Python

unknown

Dec 22, 2023

CONTENTS

Remote Syslog documention holds all documentation of the Remote Syslog products. More information can be found on https://www.remotesyslog.com/

This is a part of the masterscript: https://www.github.com/tslenter/RS

Check out the *2.1 Installation RSE core* section for further information about the installation, including how to *3. Usage CLI* or *4. Usage GUI* the project.

Note: This product is free and given without any warrenty. If you notice any fault, bug or security issue, please contact us via mail: info@remotesyslog.com.

CHAPTER

ONE

CONTENTS

1.1 1. Requirements

1.1.1 1.1 Supported operating systems

All version supported.

```
Ubuntu 20.04 LTS
Ubuntu 22.04 LTS
Debian 10.x
Debian 12.x
```

1.1.2 1.2 Tested hardware

Only for RSE or RSC.

```
Raspberry 4
```

1.1.3 1.3 Bare-metal / Virtual machine

All versions supported.

```
RSE/RSL:
1.5Ghz dual core
4 GB RAM
14GB diskspace

RSC:
1Ghz single core
2GB RAM
14GB diskspace

RSX:
2Ghz dual core
8 - 16GB RAM
50GB diskspace
```

1.2 2. Installation instuctions

1.2.1 2.1 Installation RSE core

1) To use RSE core, download the masterscript:

```
git clone https://www.github.com/tslenter/RS
```

2) Run the RSE installation

Run the script masterscript as follows:

```
cd RS
sudo ./rseinstaller
```

3) Select the following options to install the correct core:

```
Option 1 => RSE Core installation
Option 1 => Core installation
```

The installation for RSE core is now completed.

1.2.2 2.2 Installation RSC core

Required core = RSC core

1) To use RSC core, download the masterscript:

```
git clone https://www.github.com/tslenter/RS
```

2) Run the RSC installation

Run the script masterscript as follows:

```
cd RS
sudo ./rseinstaller
```

3) Select the following options to install the correct core:

```
Option 2 => RSC Core installation
Option 1 => Core installation
```

The installation for RSE core is now completed.

1.2.3 2.3 Installation RSE webinterface

1) To install the RSE webinterface, run:

```
rseinstaller
```

2) Select the following options to install the correct webinterface:

```
Option 4 => RSE webinterface installation
Option 2 => Install RSE WEB
```

The installation for RSE webinterface is now completed.

1.2.4 2.4 Installation RSC webinterface

Required core = RSE core

1) To install the RSC webinterface, run:

```
rseinstaller
```

2) Select the following options to install the correct webinterface:

```
Option 3 => RSC webinterface installation
Option 2 => Install RSC WEB
```

The installation for RSC webinterface is now completed.

1.2.5 2.5 Installation RSX webinterface

Required core = RSE core

1) To install the RSX webinterface, run:

```
rseinstaller
```

2) Select the following options to install the correct webinterface:

```
Option 5 => RSX webinterface installation
Option 2 => Install RSX WEB
```

The installation for RSX webinterface is now completed.

1.2.6 2.6 Installation RSL webinterface (Clean project)

Required core = RSE core

Remote Syslog RSL clean allows you to install a clean Laravel project for Remote Syslog.

1) To install the RSL webinterface, run:

```
rseinstaller
```

2) Select the following options to install the correct webinterface:

```
Option 6 => RSL devkit
Option 2 => RSL Clean
```

The installation for RSL webinterface is now completed.

1.2.7 2.7 Installation RSL webinterface (Backup project)

Required core = RSE core

Remote Syslog RSL backup allows you to restore a Laravel project for Remote Syslog.

1) To install the RSL webinterface, run:

```
rseinstaller
```

2) Select the following options to install the correct webinterface:

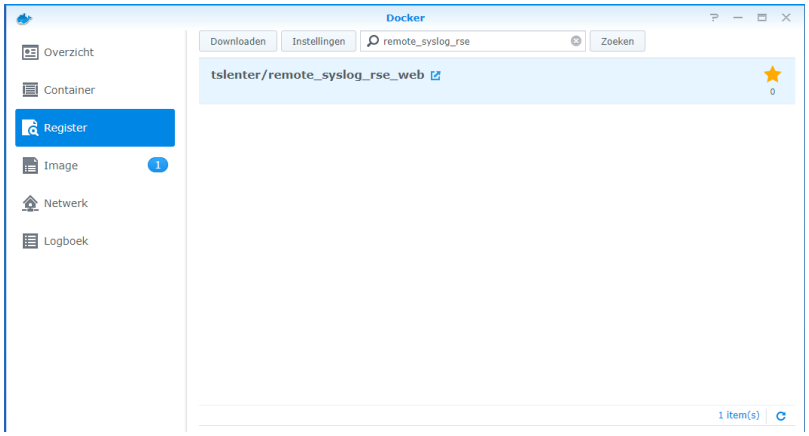
```
Option 6 => RSL devkit
Option 1 => RSL Backup
```

The installation for RSL webinterface is now completed.

1.2.8 2.8 Installation RSE for docker on Synology

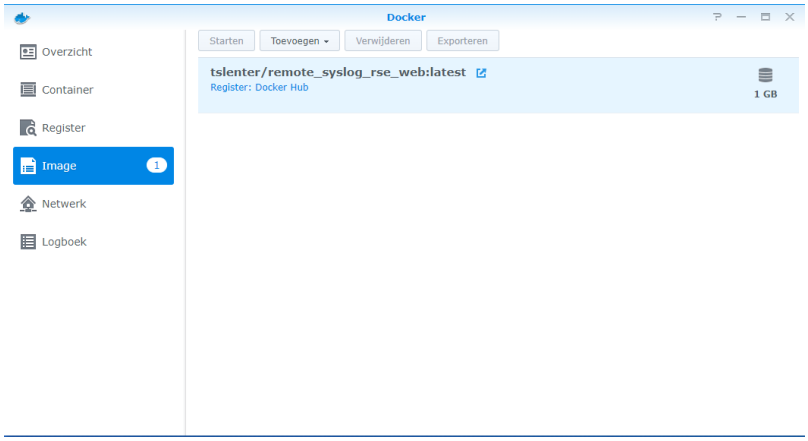
Login to the Synlogy console. Usually found: https://<nas_ip>:5000

2.8.1 Download image



Search and download as show in the image above.

2.8.2 Install image

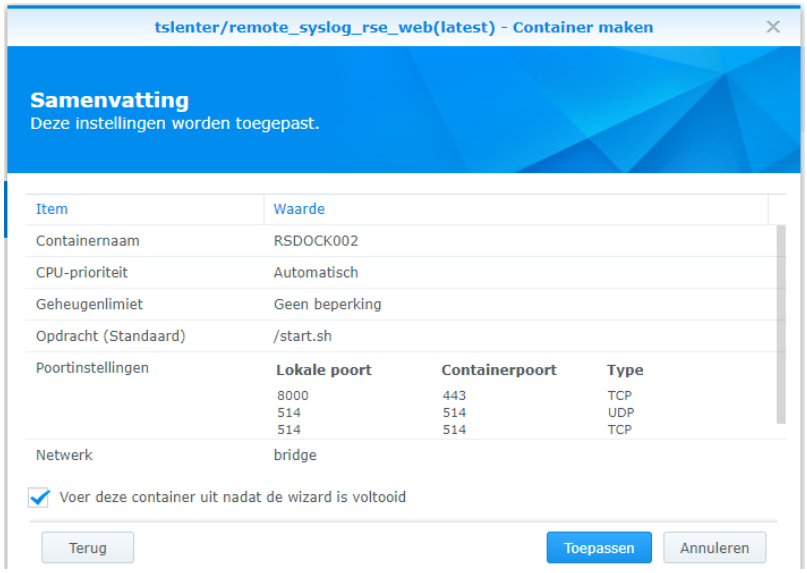


Press launch(start) as show in the image above to install the RSE image.

2.8.3 Configure network

Add the network ports as in the image above to configure the right forwarding.

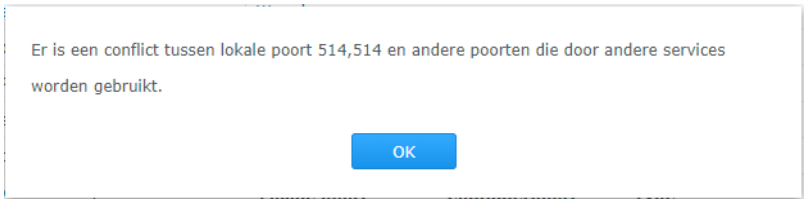
2.8.4 Check summary



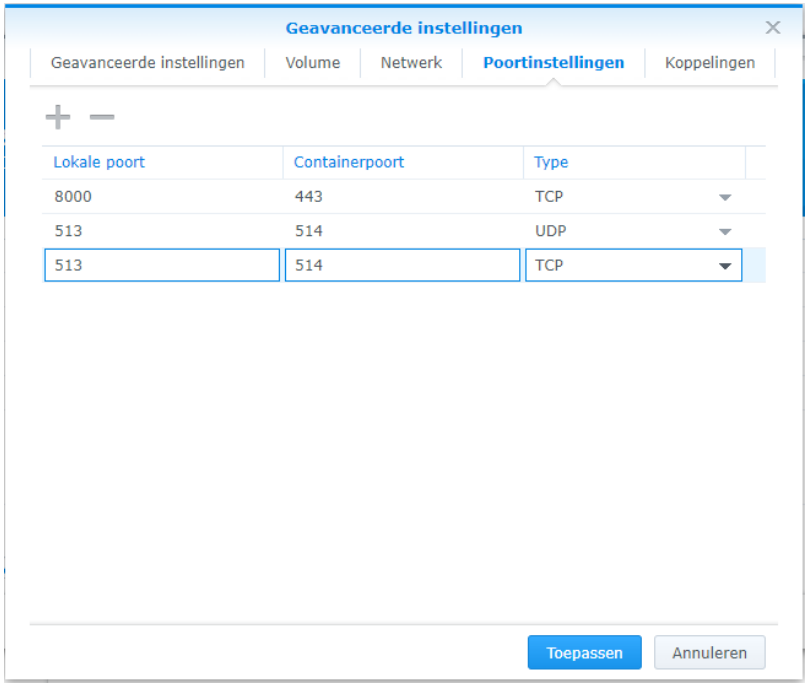
Check the setting. The image above has the correct values. The container auto-starts.

2.8.5 Known error afer apply

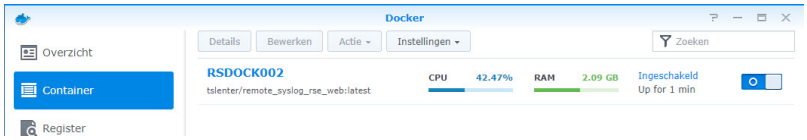
If this error is not shown to you, please skip this section.



Edit the port settings as following:



Stop the container within the Synology GUI.



After this login to the CLI of the Synology NAS and search for the following file:

Edit the file as following and change the port 513 to 514:

```
vi /volume4/@docker/containers/
↪73a981a58c43e92b8bd26226094d830d746244fa211f68e31748f2c446a963fa/hostconfig.json

{
  "Binds": [],
  "ContainerIDFile": "",
  "LogConfig": {
    "Type": "db",
```

(continues on next page)

(continued from previous page)

```
    "Config": {}
  },
  "NetworkMode": "bridge",
  "PortBindings": {
    "443/tcp": [
      {
        "HostIp": "",
        "HostPort": "8000"
      }
    ],
    "514/tcp": [
      {
        "HostIp": "",
        "HostPort": "514" <<<==
      }
    ],
    "514/udp": [
      {
        "HostIp": "",
        "HostPort": "514" <<<<==
      }
    ]
  },
```

Restart the Synology services:

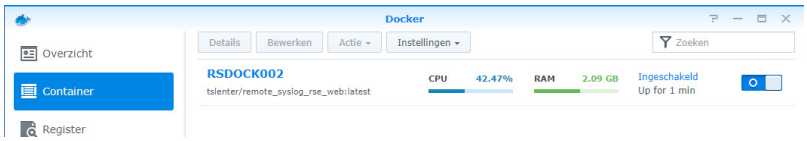
Synology version 6.x:

```
synoservice --restart pkgctl-Docker
```

Synology version 7.x:

```
sudo synopkgctl stop Docker
sudo synopkgctl start Docker
```

After this configuration start the container within the webinterface of the Synology NAS.



2.8.6 Known limitation

Due the bridge interface the source is displayed as the bridge interface.

2.8.7 Default login

The web interface is available @ https://<nas_ip>:8000. The default login is: Username "test" and the password is "Test123!"

To change the user use the reseview tool within the docker container. (CLI)

1.2.9 2.9 Docker image creation from scratch

Example is given for Remote Syslog RSE.

2.9.1 Prepare docker image

Download the ubuntu image from the registered images.

And run the following commands:

```
apt update && apt upgrade -y
apt install lsb-release wget dialog wget git apt-utils gnupg2 -y
git clone https://www.github.com/tslenter/RS
cd RS
./rseinstaller
```

Change the following file:

```
vi /etc/elasticsearch/elasticsearch.yml

add:

xpack.ml.enabled: false
```

2.9.2 Create docker image

Example code to create a docker image:

```
docker commit --message "Remote Syslog RSE for Synology ..." --author tom.
↪slenter@remotesyslog.com RSDOCK001
docker images
docker tag 86bc8a78b689 tslenter/remote_syslog_rse_web
docker login --username=<username>
docker push <username>/remote_syslog_rse_web:latest
```

1.3 3. Usage CLI

RSE CLI is usable for RSX and RSE. RSC CLI is only usable for RSC.

1.3.1 3.1 RSE viewer (rseview)

```
Usage rseview:

-h,--help                                          Display help
-s,--search <search string> <buffer>               Search through logging
-v,--view <buffer>                                 View logging
-l,--live <buffer>                                 View live logging
-ls,--livesearch <search string> <buffer>          Search through live logging
-t,--testmessage                                   Send a test message
-c,--clearlog                                      Clear log index
-p,--lifecyclepolicy                               Change lifecycle policy
-ps,--pslifecyclepolicy                            View lifecycle policy
-xi,--indexinfo                                    View indexes
-xh,--healthinfo                                   View elasticsearch health
-u,--usage                                         View disk / ram usage
```

3.1.1 Display help

```
rseview -h
```

Displays help menu with all available options.

3.1.2 Display logging with search string

```
rseview -s mysearch 90
```

Displays logging with the given search string. Output will be given in the console. Buffer of 90 gives the latest 90 results.

3.1.3 Display logging

```
rseview -v 90
```

Displays the latest logging. Output will be given in the console. Buffer of 90 gives the latest 90 results. Buffer is default 50.

3.1.4 Display live logging

```
rseview -l 90
```

Displays live logging. Output will be given in the console. Buffer of 90 gives the latest 90 results. Buffer is default 50.

3.1.5 Display live logging with search string

```
rseview -ls mysearch 90
```

Displays live logging with the given search string. Output will be given in the console. Buffer of 90 gives the latest 90 results. Buffer is default 50.

3.1.6 Generate test message

```
rseview -t
```

Generates a test message. Run "rseview -s test" to check if it was successfull.

3.1.7 Clear all logging

```
rseview -c
```

Clears all logging. Output will be given in the console.

3.1.8 Change policy

```
rseview -p
```

Sets a new lifecycly plocicy for the elasticsearch remote syslog index. Data is given in day and gigabyte. Output will be given in the console.

3.1.9 Display policy

```
rseview -ps
```

Displays the lifecycle policy. Output will be given in the console.

3.1.10 Display index / shard info

```
rseview -xi
```

Displays index / shard info. Output will be given in the console.

3.1.11 Display cluster / server info

```
rseview -xh
```

Displays cluster / server info. Output will be given in the console.

3.1.12 Display usage

```
rseview -u
```

Displays disk and RAM info. Output will be given in the console.

3.1.13 Authentication for RSEVIEW

If you setup authentication for Elasticsearch then the RSEVIEW gets a update with the username and password as well. You need to change the following files:

File 1: /usr/bin/rseview

```
Change with the username and password:

USERNAME=
PASSWORD=
```

File 2: /opt/RSEVIEW/rs_init.php

```
Change with the username and password:

'user' => '',
'pass' => ''
```

The default login is nothing. When changed, the login username is probably “elastic”

3.1.14 Filter on date and string

To filter a date and a string, use the following example:

```
rseview -s "R_ISODATE:2022-08-15T13\\:56 AND myswitch" 400
```

To filter the date we use the R_ISODATE field.

1.3.2 3.2 RSC viewer (rsview)

```
Usage rsview:

-h,--help                          Display help
-s,--search <search string>        Search through logging
-v,--view                          View logging
-l,--live                          View live logging
-ls,--livesearch <search string>   Search through live logging
```

(continues on next page)

(continued from previous page)

```
-t,--testmessage                    Send a test message
-c,--clearlog                       Clear total log archive
```

3.2.1 Display help

```
rsview -h
```

Displays help menu with all available options.

3.2.2 Display logging with search string

```
rsview -s mysearch
```

Displays logging with the given search string. Output will be given in the console.

3.2.3 Display logging

```
rsview -v
```

Displays the latest logging. Output will be given in the console.

3.2.4 Display live logging

```
rsview -l
```

Displays live logging. Output will be given in the console.

3.2.5 Display live logging with search string

```
rsview -ls
```

Displays live logging with the given search string. Output will be given in the console.

3.2.6 Generate test message

```
rsview -t
```

Generates a test message. Run “rsview -s test” to check if it was successfull.

3.2.7 Clear all logging

```
rsview -c
```

Clears all logging. Output will be given in the console.

1.3.3 3.3 RSE user management (rseuser)

```
Usage rseuser:

Please use the command as: rseuser <username> <rm or add> <web-only>
```

3.3.1 Add user

```
rseuser tom add web-only
```

Creates a user tom for the webinterface only. Drop the web-only option to setup a user for CLI.

3.3.2 Remove user

```
rseuser tom rm
```

Removes the user tom.

1.3.4 3.4 RSC user management (rsuser)

```
Usage rsuser:

Please use the command as: rsuser <username> <rm or add> <web-only>
```

3.4.1 Add user

```
rsuser tom add web-only
```

Creates a user tom for the webinterface only. Drop the web-only option to setup a user for CLI.

3.4.2 Remove user

```
rsuser tom rm
```

Removes the user tom.

1.3.5 3.5 Python module

Remote Syslog rslogger can be used to write important lines of informational logging from a python script to a remote syslog server. We found it usefull as we run multiple scripts on different hosts. With this we track the given info on a central / remote server. Example use case: automation scripts for device configuration.

3.5.1 Requirements

- Remote Syslog core or other syslog listener must be running as minimum
- Python script below has the same path as the running python script

3.5.2 Installation

Install the python socket module using the following command:

```
pip install socket
```

Get a local copy of this repo:

```
git clone https://github.com/tslenter/rslogger
cd rslogger
#On Windows:
copy rslogger <Directory of the project>
#On Linux
cp rslogger <Directory of the project>
```

3.5.3 Example with Cisco DNA Controller

The following is a demo example that extracts data from a Cisco DNA controller and sends the data string to a syslog socket:

```
import requests
import os
from requests.auth import HTTPBasicAuth
import urllib3
import argparse
from rslogger import syslog
from rslogger import fcl
from rslogger import lvl

#Disable HTTPS validation
urllib3.disable_warnings()

#Set variables to None
hostname = None
username = None
password = None

#Create HTTP header
headers = {
              'content-type': "application/json",
```

(continues on next page)

(continued from previous page)

```
            'x-auth-token': ""
        }

#Global information
print('Running from directory: ', os.getcwd())

#Add arguments
parser = argparse.ArgumentParser()
parser.add_argument('-n', '--hostname',  help='Enter a hostname or ip of the Cisco DNA␣
↪Controller', required=True)
parser.add_argument('-u','--username', help='Add a username', required=True)
parser.add_argument('-p', '--password', help='Add a password', required=True)
args = parser.parse_args()

#Extract variables from namespace to global
globals().update(vars(args))

#Generate token for DNA Controller
def dnac_login(host, passwrd, user):
    # Generate token
    BASE_URL = 'https://' + host
    AUTH_URL = '/dna/system/api/v1/auth/token'
    USERNAME = user
    PASSWORD = passwrd

    response = requests.post(BASE_URL + AUTH_URL, auth=HTTPBasicAuth(USERNAME, PASSWORD),
↪ verify=False)
    token = response.json()['Token']
    return token

#Extract data from DNA controller
def network_device_list(token, host):
    url = "https://" + host + "/api/v1/network-device"
    headers["x-auth-token"] = token
    response = requests.get(url, headers=headers, verify=False)
    data = response.json()
    for item in data['response']:
        #Feel free to list more information: item["hostname"],item["platformId"],item[
↪"softwareType"],item["softwareVersion"],item["upTime"], item["serialNumber"], item[
↪"managementIpAddress"]
        message = str("hostname: ")+item["hostname"]
        syslog(message, level=lvl['notice'], facility=fcl['log_audit'], host='172.16.201.
↪2', port=514)

#Login to DNA Controller
if hostname or username or password != None:
    print("Started session on: " + hostname)
    print("Started session with user: " + username)
    login = dnac_login(hostname, password, username)
    network_device_list(login, hostname)
else:
    print("Did you use the parameters to run this command?")
```

3.5.4 Available facility

kern, user, mail, daemon, auth, syslog, lpr, news, uucp, cron, authpriv, ftp, ntp, log_audit, log_alert, clock_daemon, local0, local1, local2, local3, local4, local5, local6, local7

3.5.5 Available levels

emerg, alert, crit, err, warning, notice, info, debug

3.5.6 Most basic code (Example)

```
from rslogger import syslog
from rslogger import fcl
from rslogger import lvl
```

Run test message to localhost (a syslog server is needed)

```
syslog()
```

Expected output:

```
Jul  5 17:02:21 localhost daemon: notice: Test is RS test message to localhost
```

Run with variables:

```
message=str('Hello world')
syslog(message, level=lvl['alert'], facility=fcl['daemon'], host='172.16.201.2',␣
↪port=514)
```

Expected output (syslog server):

```
Jul  5 17:02:21 comp0001.remotesyslog.com rslogger: daemon: alert: rslogger_output:␣
↪Hello world
```

1.3.6 3.6 Remote Syslog Programmer

Remote Syslog Programmer is a ssh connector written in python to configure device with SSH support. This connector can be used on multiple vendors. Tested for Ubiquiti and Cisco devices.

Capabilities: Configure multiple devices with the same configuration. All output will be written to a plain text file.

Created for: To update logging configuration for all network devices of the same type. Can be used for other configurations!

3.6.1 Installation

Copy to repo to the local machine:

```
git clone https://www.github.com/tslenter/RSPROGRAMMER
cd RSPROGRAMMER
```

3.6.2 Usage

Run as single cli command with multiple remote commands do:

```
python ssh_connect.py -n 172.16.9.1,172.16.10.1 -u <username> -p <strong_pw> -f commands.
↪txt
```

Or for 1 host:

```
python ssh_connect.py -n 172.16.9.1 -u <username> -p <strong_pw> -f commands.txt
```

Run as single cli command with a single commands do:

```
python ssh_connect.py -n 172.16.9.1,172.16.10.1 -u <username>  -p <strong_pw> -f "sh int␣
↪status"
```

Or for 1 host:

```
python ssh_connect.py -n 172.16.9.1 -u <username>  -p <strong_pw> -f "sh int status"
```

Run in interactive mode:

```
python ssh_connect.py

================================
Interactive mode is loaded!
Enter switch: mysw001,mysw002
Enter username: <username>
Enter password: <strong_pw>
Enter filename or press enter for single command option: <enter file name like command.
↪txt or press enter>
If you pressed enter the next question appears:
Enter command: <Type command>
```

The output of the commands will be written to: output.txt.

All options for ssh_connect.py:

```
python ssh_connect.py -h

Script is created by T.Slenter
The switches input is as following: hostname or ip,hostname or ip,hostname or ip
Running from directory:  F:\ssh_connector\ssh_connector
usage: ssh_connect.py [-h] [-n HOST] [-u USERNAME] [-p PASSWORD] [-s SINGLECOMMAND] [-f␣
↪FILE]

optional arguments:
```

(continues on next page)

(continued from previous page)

```
  -h, --help                                Show this help message and exit
  -n HOST, --host HOST                      Enter a hostname or ip, multiple hostname and␣
↪ips are supported use seperator=,
  -u USERNAME, --username USERNAME          Add a username
  -p PASSWORD, --password PASSWORD          Add a password
  -s SINGLECOMMAND, --singlecommand         SINGLECOMMAND      Enter a single command
  -f FILE, --file FILE                      Add file with commands
```

1.3.7 3.7 RSEDUMPER

RSEDUMPER is a small tool that can dump the default RSE index with color style.

3.7.1 Installation

Copy to repo to the local machine:

```
git clone https://www.github.com/tslenter/RSEDUMPER
cd RSEDUMPER
cp rsedumper /usr/bin/
```

3.7.2 Usage

Run as single cli command with multiple remote commands do:

```
ubuntu@rssyslog001:~$rsedumper

#################################################
#Remote Syslog Elasticsearch Dumper             #
#More information: https://www.remotesyslog.com #
#Remote Syslog dumper for Elasticsearch         #
#Version: RSEDUMPER 0.1                         #
#URL: https://github.com/tslenter/RSEDUMPER     #
#Donation: https://github.com/tslenter/RS       #
#################################################

Usage rseview:

-h,--help               Display help
-c,--color              Dump default RSE index in color
-n,--nocolor            Dump default RSE index without color
```

Start the dump with:

```
without color:

rsedumper -n

or with color:

rsedumper -c
```

1.4 4. Usage GUI

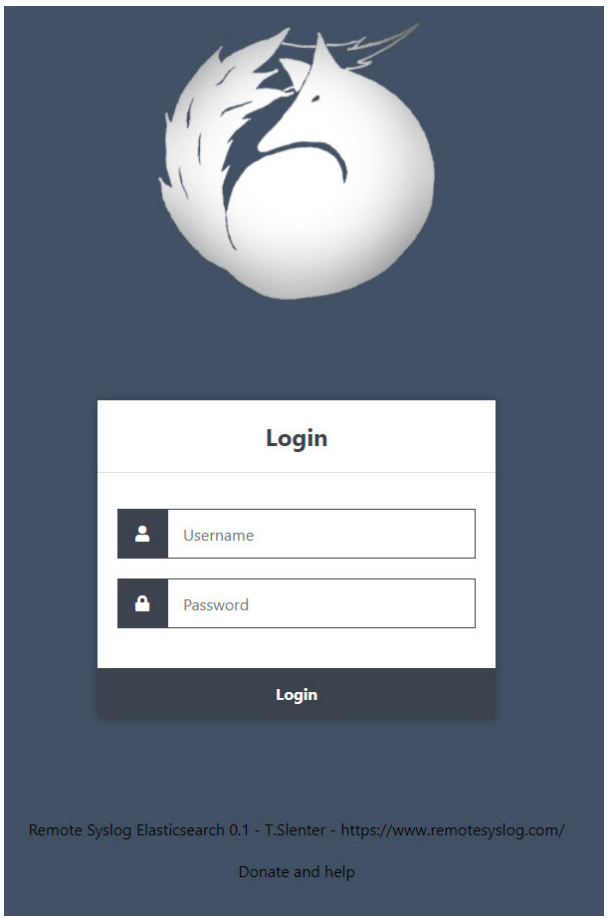
1.4.1 4.1 GUI Requirements

```
RSE Core is required for RSX and RSE.
RSC Core is required for for RSC.
```

1.4.2 4.2 RSE GUI usage

Documentation is build for RSE version 0.1.

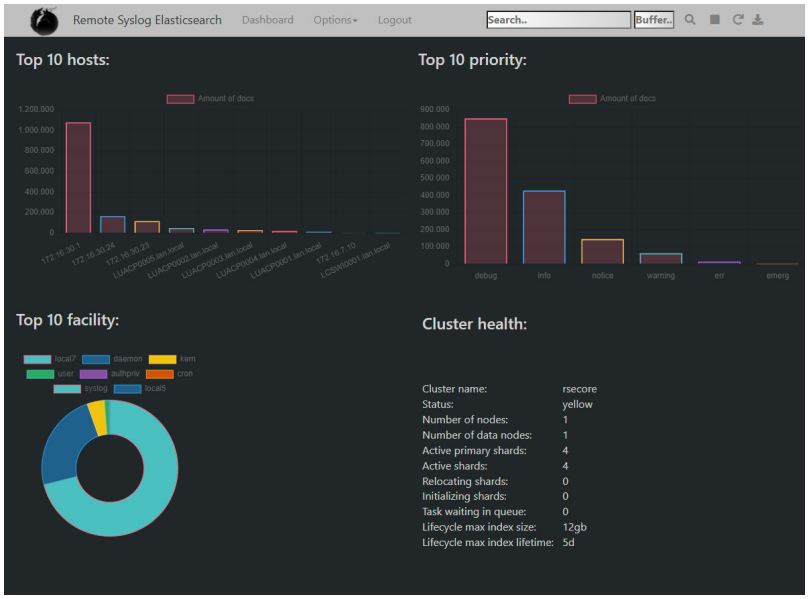
4.2.1 RSE Login



Above image shows the default login. which can be found with the following url: https://<ip-of-server>/. To login use the account what was used during the installation. The password requires the following:

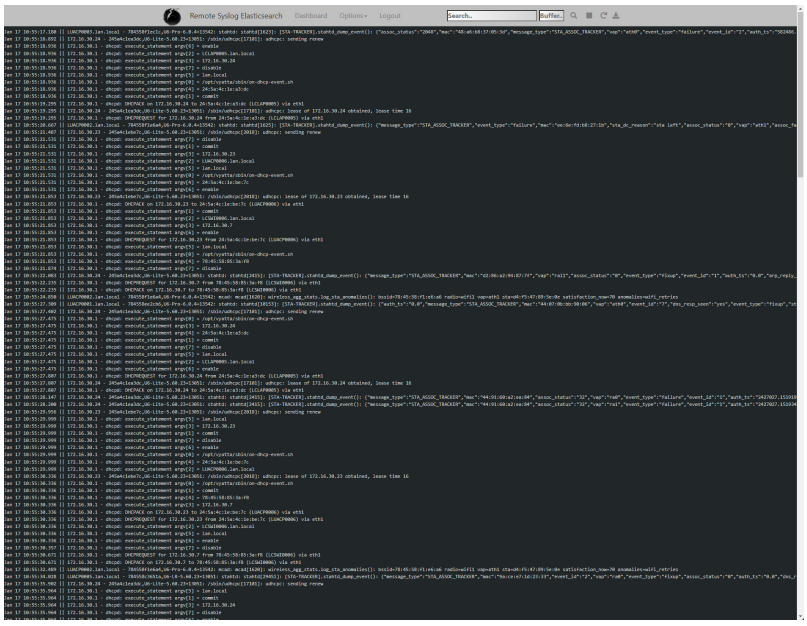
```
- Minimum of 4 characters
- Minimum of 1 capital letter
- Minimum of 1 special characters (!, @, #, $, %, ^, &, *, _, =, +, -)
```

4.2.2 RSE dashboard



Above image shows the dashboard and it contains the top 10 stats with a heath indicator of the elasticsearch engine.

4.2.3 RSE text viewer

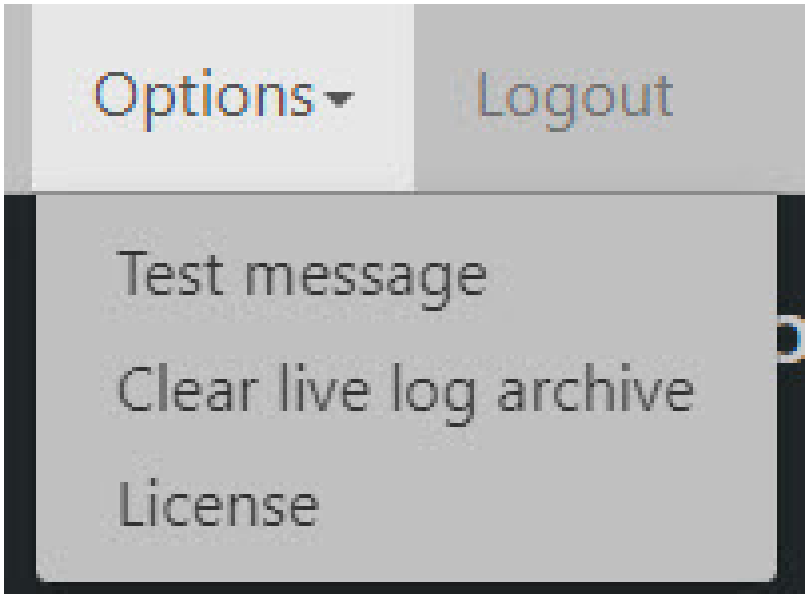


The image above gives the default view after the login. Here you can search end view live logging.

4.2.4 RSE menu

The image above displays the menu. Select "Dashboard" to view the dashboard page. "Options" holds some more additions configurations. "Logout" logs the connected user out.

4.2.5 RSE options



Within the option menu there are 3 option. "Test message" sends a UPD and TCP test message to the system. "Clear live log archive" clears all logging from the server. "License" redirects you to the license page of Remote Syslog.

4.2.6 RSE searchbar

The searchbar allows to search live logging on fields, regex and on text strings. It buffer field allows to give any number between 0 and 3000 to limit or extend search results.

The buttons:

1) The "search" button allows you to view the live logging with ot without a searchstring or buffer value.
2) The "stop" button stop the live logging with or without a searchsting or buffer value.
3) The "redo" button loads the default settings and the live logging strats scolling without buffer and searchsting.
4) The "download" button exports the text to a HTML file.

4.2.7 RSE searchbar example searchstrings

By default 3 fields are used by RSE. These are:

1) MESSAGE
2) DATE
3) HOST_FROM

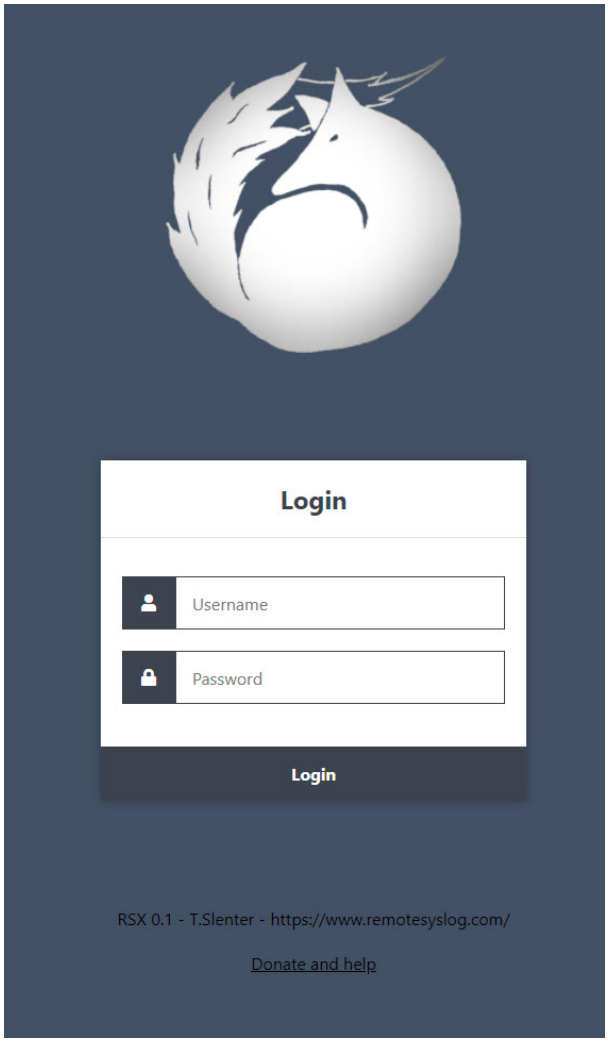
Table 1: Search options

What:	Command:	Tested:
MAC address:	\'24\\:5a\\:4c\\:1e\\:a3\\:dc\'	V
Search on a field:	MESSAGE: com*	V
Use wildcard:	com*	V

1.4.3 4.3 RSX GUI usage

Documentation is build for RSX version 0.1.

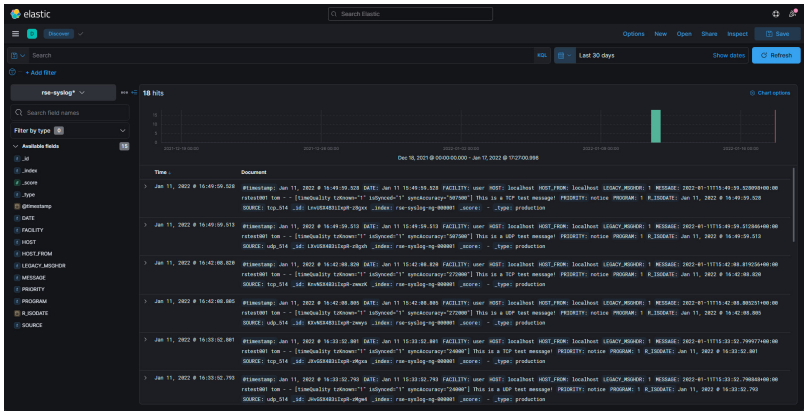
4.3.1 RSX Login



Above image shows the default login. which can be found with the following url: https://<ip-of-server>/. To login use the account what was used during the installation. The password requires the following:

```
- Minimum of 4 characters
- Minimum of 1 capital letter
- Minimum of 1 special characters (!, @, #, $, %, ^, &, *, _, =, +, -)
```

4.3.2 RSX dashboard



Above image shows the dashboard of RSX. It is the elasticsearch interface “Kibana”. More information can be found here: https://www.elastic.co/

4.3.3 RSX logout

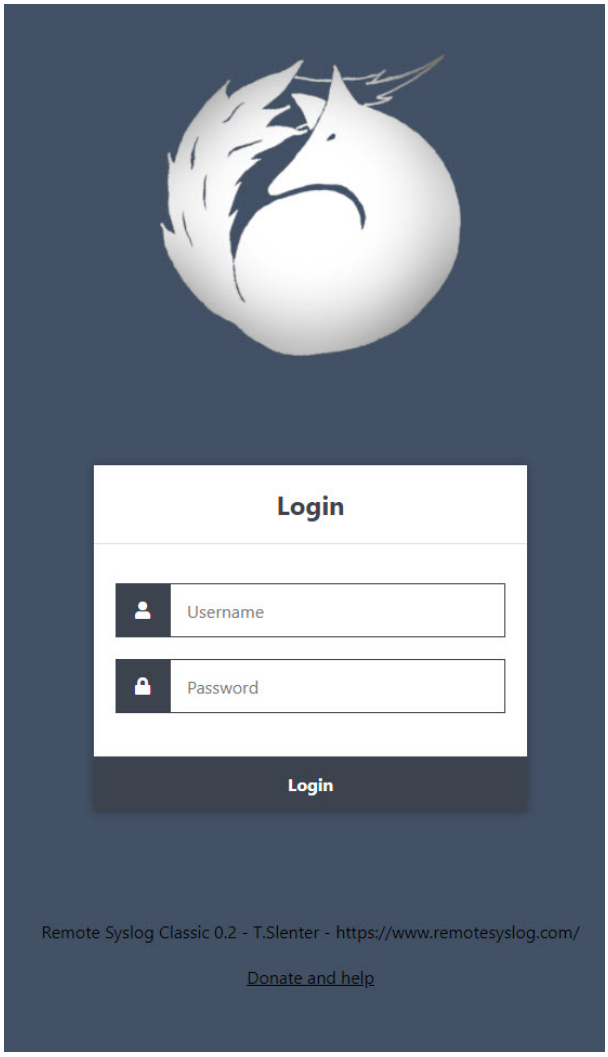
Use the following url to logout:

```
https://<ip or dns>/logout
```

1.4.4 4.4 RSC GUI usage

Documentation is build for RSC version 0.2.

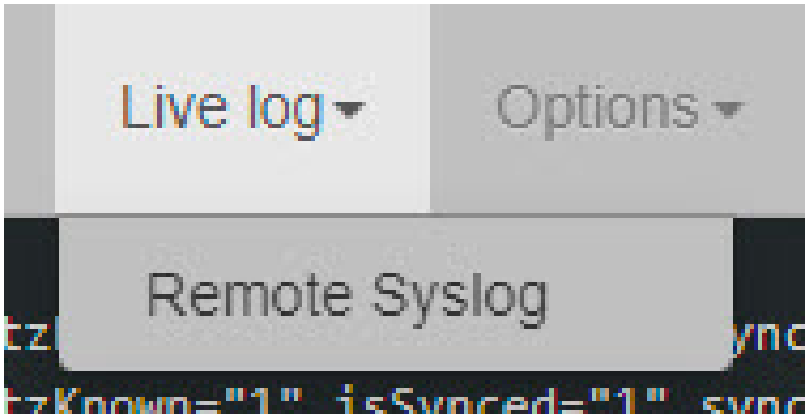
4.4.1 RSC Login



Above image shows the default login. which can be found with the following url: https://<ip-of-server>/. To login use the account what was used during the installation. The password requires the following:

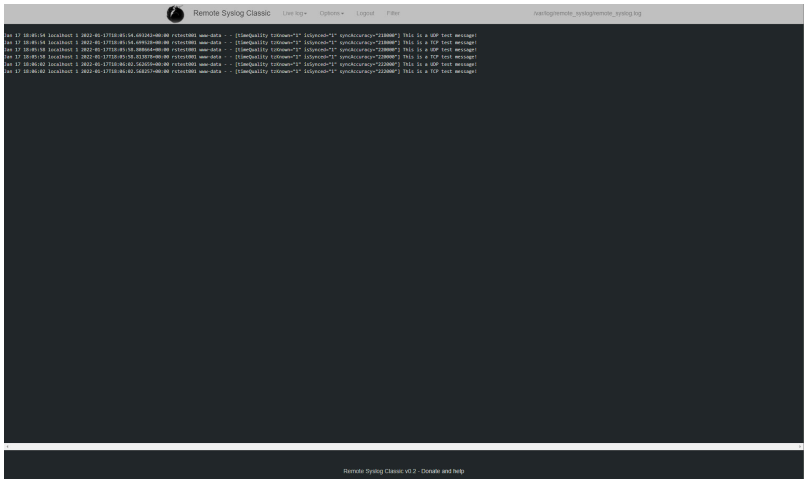
```
- Minimum of 4 characters
- Minimum of 1 capital letter
- Minimum of 1 special characters (!, @, #, $, %, ^, &, *, _, =, +, -)
```

4.4.2 RSC livelog selection



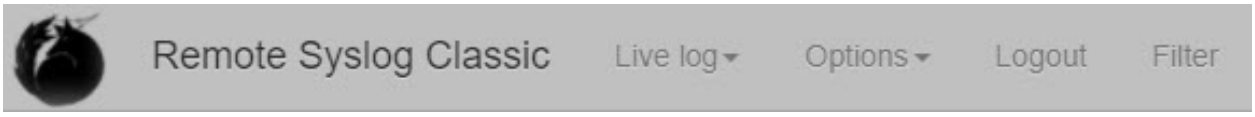
When using PHP programming skills there is a option to add addition text files for logging. Check out GitHub for the sourcecode.

4.4.3 RSC text viewer



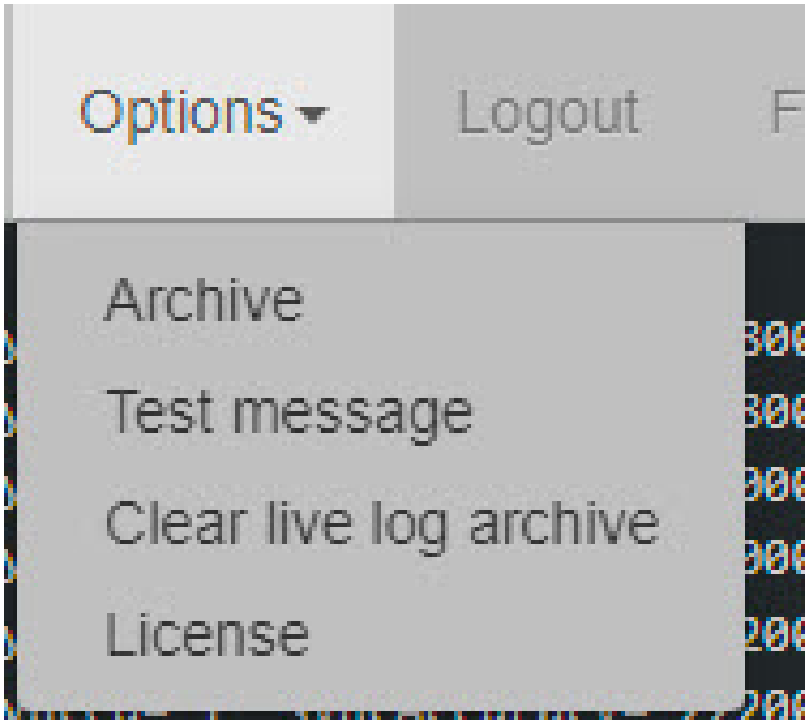
The image above gives the default view after the login. Here you can search end view live logging.

4.4.4 RSC menu



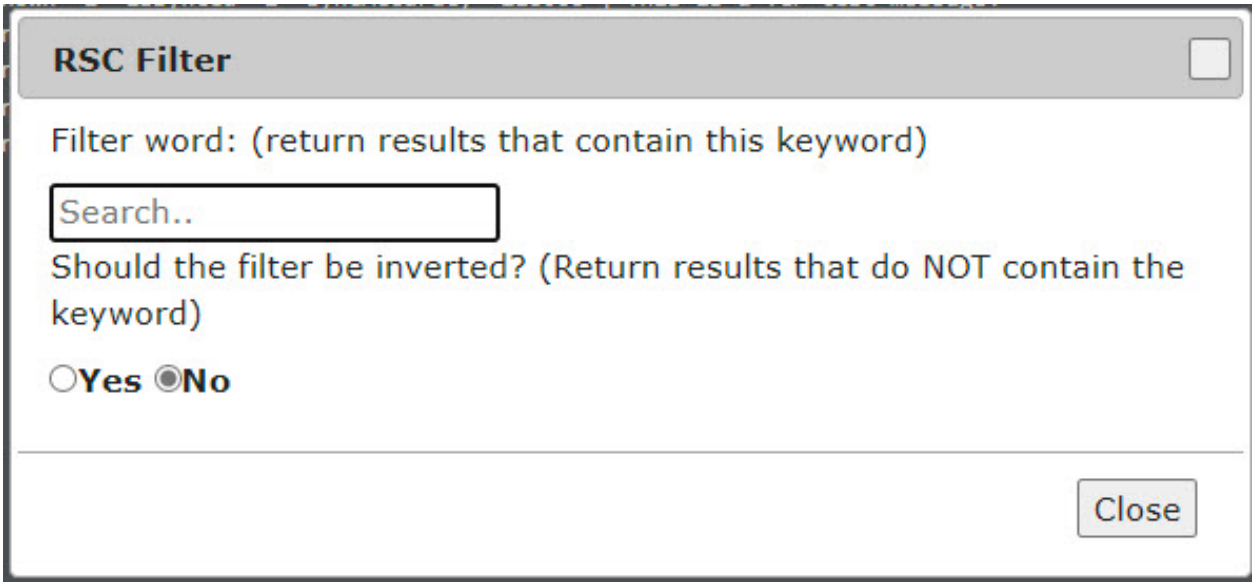
The image above displays the menu. Select “Live log” to view the Remote Syslog log file. “Options” holds some more additions configurations. “Logout” logs the connected user out. “Filter” gives a pop-up with the filter options.

4.4.5 RSC options



Within the option menu there are 4 option. “Archive” opens the archive directory to donwload historic data. “Test message” sends a UPD and TCP test message to the system. “Clear live log archive” clears all logging from the server. “License” redirects you to the license page of Remote Syslog.

4.4.6 RSC filter



The filter allows to search live logging on text strings. There is a possibily to reverse the search. Press “Close” to apply the filter.

1.5 5. Deinstallation

1.5.1 5.1 Deinstallation RSE core

Make sure you removed the RSX/RSE webinterface first.

1) To remove the RSE core, run:

```
rseinstaller
```

2) Select the following options to remove the correct core:

```
Option 1 => RSE Core installation
Option 2 => Core removal
```

The removal for RSE core is now completed.

1.5.2 5.2 Deinstallation RSC core

Make sure you removed the RSC webinterface first.

1) To remove the RSC core, run:

```
rseinstaller
```

2) Select the following options to remove the correct core:

```
Option 2 => RSC Core installation
Option 2 => Core removal
```

The removal for RSC core is now completed.

1.5.3 5.3 Deinstallation RSE webinterface

1) To remove the RSE webinterface, run:

```
rseinstaller
```

2) Select the following options to remove the correct webinterface:

```
Option 4 => RSE webinterface installation
Option 3 => Remove RSE WEB
```

The deinstallation for RSE webinterface is now completed.

1.5.4 5.4 Deinstallation RSC webinterface

Required core = RSE core

1) To remove the RSC webinterface, run:

```
rseinstaller
```

2) Select the following options to remove the correct webinterface:

```
Option 3 => RSC webinterface installation
Option 3 => Remove RSC WEB
```

The deinstallation for RSC webinterface is now completed.

1.5.5 5.5 Deinstallation RSX webinterface

Required core = RSE core

1) To remove the RSX webinterface, run:

```
rseinstaller
```

2) Select the following options to remove the correct webinterface:

```
Option 5 => RSX webinterface installation
Option 3 => Remove RSX WEB
```

The deinstallation for RSX webinterface is now completed.

1.5.6 5.6 Deinstallation RSL webinterface (Any project)

Required core = RSE core

Remote Syslog RSL clean allows you to remove a clean Laravel project for Remote Syslog.

1) To remove the RSL webinterface, run:

```
rseinstaller
```

2) Select the following options to remove the correct webinterface:

```
Option 6 => RSL devkit
Option 3 => RSL Removal
```

The deinstallation for RSL webinterface is now completed.

1.5.7 5.7 Deinstallation rseinstaller

1) To remove the rseinstaller command, run:

```
rseinstaller
```

2) Select the following options to remove rseinstaller:

```
Option 8 => RSEINSTALLER removal
```

The deinstallation for rseinstaller is now completed.

1.6 6. Upgrade

1.6.1 6.1 Upgrade RSE core

1) To upgrade the RSE core, run:

```
rseinstaller
```

2) Select the following options to upgrade the correct core:

```
Option 1 => RSE Core installation
Option 3 => Core upgrade
```

The upgrade for RSE core is now completed.

1.6.2 6.2 Upgrade RSC core

1) To upgrade the RSC core, run:

```
rseinstaller
```

2) Select the following options to upgrade the correct core:

```
Option 2 => RSC Core installation
Option 3 => Core upgrade
```

The upgrade for RSC core is now completed.

1.6.3 6.3 Upgrade RSE webinterface

1) To upgrade the RSE webinterface, run:

```
rseinstaller
```

2) Select the following options to upgrade the correct webinterface:

```
Option 4 => RSE webinterface installation
Option 1 => Upgrade RSE WEB
```

The upgrade for RSE webinterface is now completed.

1.6.4 6.4 Upgrade RSC webinterface

Required core = RSE core

1) To upgrade the RSC webinterface, run:

```
rseinstaller
```

2) Select the following options to upgrade the correct webinterface:

```
Option 3 => RSC webinterface installation
Option 1 => Upgrade RSC WEB
```

The upgrade for RSC webinterface is now completed.

1.6.5 6.5 Upgrade RSX webinterface

Required core = RSE core

1) To upgrade the RSX webinterface, run:

```
rseinstaller
```

2) Select the following options to upgrade the correct webinterface:

```
Option 5 => RSX webinterface installation
Option 1 => Upgrade RSX WEB
```

The upgrade for RSX webinterface is now completed.

1.6.6 6.6 Upgrade RSL webinterface (Any project)

Required core = RSE core

Remote Syslog RSL clean allows you to upgrade a clean Laravel project for Remote Syslog.

1) To upgrade the RSL webinterface, run:

```
rseinstaller
```

2) Select the following options to remove the correct webinterface:

```
Option 6 => RSL devkit
Option 3 => RSL Removal
```

3) Reinstall a project from backup, run:

```
rseinstaller
```

4) Select the following options to install the correct webinterface:

```
Option 6 => RSL devkit
Option 1 => RSL Backup
```

The upgrade for RSL webinterface is now completed.

1.6.7 6.7 Upgrade from legacy Remote Syslog

Manual remove Remote Syslog 1.x with the following bash script:

```
echo "File is only present if local syslog is activated"
rm -rf /etc/syslog-ng/conf.d/99-remote-local.conf
echo "Remove configuration files"
rm -rf /etc/syslog-ng/conf.d/99-remote.conf
rm -rf /etc/logrotate.d/remotelog
rm -rf /etc/colortail/conf.colortail
rm -rf /opt/remotesyslog
echo "Remove binary files"
rm -rf /usr/bin/rsview
rm -rf /usr/bin/rsinstaller
echo "Removing legacy GUI website ..."
rm -rf /var/www/html/favicon.ico
rm -rf /var/www/html/index.php
rm -rf /var/www/html/indexs.php
rm -rf /var/www/html/jquery-latest.js
rm -rf /var/www/html/loaddata.php
echo "Remove packages ..."
apt -y purge apache2 apache2-utils php libapache2-mod-php syslog-ng colortail
apt -y autoremove
echo "Reinstall rsyslog"
apt -y install rsyslog
```

After the removal of Remote Syslog 1.x, install the new RSX or RSC. The old syslog data is still available within the log folder /var/log/remote_syslog/.

More information over Remote Syslog 1.x: https://github.com/tslenter/Remote_Syslog

1.6.8 6.8 Upgrade to new distrobution

Example: Upgrade from Ubuntu 18.04 to 20.04

This holds a upgrade to a new Ubuntu version and some known issues from that time. Those are fixed now.

Upgrade commands:

```
apt update && sudo apt upgrade
```

You probably run in a syslog-ng rdkafka error. This will stop the installation. Therefore we added “apt install -f”. This only effects version 3.27.1 and was fixed in 3.27.1-2.

```
apt install -f
reboot
apt install update-manager-core
do-release-upgrade -d
```

It appears that the package “syslog-ng-mod-rdkafka” has some conflics with the core configuration, If you run in this error, try to uninstall this package:

```
#This only effects version 3.27.1 and was fixed in 3.27.1-2.
apt remove syslog-ng-mod-rdkafka
```

After the upgrade there is a issue with the Apache2 configuration: Edit the following file: /etc/apache2/mods-enabled/php7.2.load and change:

```
-LoadModule php7_module /usr/lib/apache2/modules/libphp7.2.so
+LoadModule php7_module /usr/lib/apache2/modules/libphp7.4.so
```

Check to /var/log/syslog for errors. We found 2 errors and this depends on which platform you run the server. Error 1 || DNS message:

```
Apr 30 20:56:22 lusysl003 systemd-resolved[923]: Server returned error NXDOMAIN,␣
↪mitigating potential DNS violation DVE-2018-0001, retrying transaction with reduced␣
↪feature level UDP
```

Recreate symlink will fix this issue:

```
ln -sfn /run/systemd/resolve/resolv.conf /etc/resolv.conf
```

or

```
rm /etc/resolv.conf
ln -s /run/systemd/resolve/resolv.conf /etc/resolv.conf
```

Error 2 || If you run the server on ESXi you get the following error:

```
Apr 30 12:47:53 plisx001.lan.local multipathd[856]: sdb: add missing path
Apr 30 12:47:53 plisx001.lan.local multipathd[856]: sdb: failed to get udev uid: Invalid␣
↪argument
Apr 30 12:47:53 plisx001.lan.local multipathd[856]: sdb: failed to get sysfs uid:␣
↪Invalid argument
Apr 30 12:47:53 plisx001.lan.local multipathd[856]: sdb: failed to get sgio uid: No such␣
↪file or directory
```

Edit the following file /etc/multipath.conf to fix this issue:

```
+blacklist {
+    device {
+        vendor "VMware"
+        product "Virtual disk"
+    }
+}
```

After that restart the deamon:

```
systemctl restart multipath-tools
```

Reactivate repo:

```
wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | apt-key add -
apt-get install apt-transport-https -y
echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main" | tee -a /etc/apt/
↪sources.list.d/elastic-7.x.list
echo "deb https://artifacts.elastic.co/packages/oss-7.x/apt stable main" | tee -a /etc/
↪apt/sources.list.d/elastic-7.x.list

wget -qO - https://cloud.remotesyslog.com/xUbuntu_18.04/Release.key | /usr/bin/apt-key␣
```

(continues on next page)

(continued from previous page)

```
↪add -
echo deb https://cloud.remotesyslog.com/xUbuntu_18.04 ./ > /etc/apt/sources.list.d/
↪syslog-ng.list
apt update
apt install syslog-ng-mod-snmp syslog-ng-mod-freetds syslog-ng-mod-json syslog-ng-mod-
↪mysql syslog-ng-mod-pacctformat syslog-ng-mod-pgsql syslog-ng-mod-snmptrapd-parser␣
↪syslog-ng-mod-sqlite3
sudo apt autoremove
```

1.6.9 6.9 Preparations for Elastic 8.x

Preparing for Elastic 8.x we have the following additional configuration. As we currently testing and validating the config, we provide the following configuration:

Create selfsinged certificate:

```
/usr/share/elasticsearch/bin/elasticsearch-certutil ca
/usr/share/elasticsearch/bin/elasticsearch-certutil cert --ca elastic-stack-ca.p12
```

Generate passwords for all Elastic users:

```
/usr/share/elasticsearch/bin/elasticsearch-setup-passwords interactive
```

Edit xpack security option within Elastic. The configuration below is tested for a cluster.

Master node:

```
xpack.security.enabled: true
xpack.security.authc.api_key.enabled: true
xpack.security.transport.ssl.enabled: true
xpack.security.transport.ssl.verification_mode: certificate
xpack.security.transport.ssl.client_authentication: required
xpack.security.transport.ssl.keystore.path: /etc/elasticsearch/elastic-certificates.p12
xpack.security.transport.ssl.truststore.path: /etc/elasticsearch/elastic-certificates.p12
cluster.name: syslog
node.name: syslog01
node.roles: [ master, data ]
network.host: 0.0.0.0
http.port: 9200
transport.port: 9300
discovery.seed_hosts:
    - 10.10.10.99
```

Data node:

```
xpack.security.enabled: true
xpack.security.authc.api_key.enabled: true
xpack.security.transport.ssl.enabled: true
xpack.security.transport.ssl.verification_mode: certificate
xpack.security.transport.ssl.client_authentication: required
xpack.security.transport.ssl.keystore.path: /etc/elasticsearch/elastic-certificates.p12
xpack.security.transport.ssl.truststore.path: /etc/elasticsearch/elastic-certificates.p12
```

(continues on next page)

(continued from previous page)

```
cluster.name: syslog
node.name: syslog02
#node.master: false
#node.data: true
node.roles: [ data ]
network.host: 0.0.0.0
http.port: 9200
#transport.tcp.port: 9300
transport.port: 9300
#discovery.zen.ping.unicast.hosts: ["10.10.10.99"]
#discovery.zen.minimum_master_nodes: 2
discovery.seed_hosts:
   - 10.10.10.99
```

Update Kibana configuration:

```
elasticsearch.username: "kibana_system"
elasticsearch.password: "MY_PASSWORD"
server.rewriteBasePath: true
server.basePath: "/kibana"
```

Update Syslog-NG configuration with username and password in the URL:

```
#Update RSE configuration
destination d_http {
 elasticsearch-http(
  frac_digits(3)
  index("rsx-routingandswitching")
  type("production")
  url("http://my_username:my_password@localhost:9200/_bulk")
  persist-name("Default RSE log")
  template("$(format-json --scope rfc5424 --scope dot-nv-pairs --scope nv-pairs --key R_
↪ISODATE @timestamp=${R_ISODATE})"));
};
```

Change the Apache2 reverse configuration for port 80 and 443:

```
<Location /kibana>
    Define CREDENTIALS my_username:my_password
    RequestHeader set Authorization "expr=Basic %{base64:${CREDENTIALS}}"
</Location>
```

1.6.10 6.10 Ubuntu upgrade policy

For Ubuntu we only test the latest LTS version. At the time of writing this is 20.04 LTS. The next release will be 22.04 LTS.

1.7 7. Additional configuration

1.7.1 7.1 Active Directory integration via PAM

Run commands as root:

```
su -
```

Upgrade distro:

```
apt-get update && apt upgrade -y
```

Install packages:

```
apt-get install realmd packagekit sssd-tools sssd libnss-sss libpam-sss adcli oddjob␣
↪oddjob-mkhomedir adcli samba-common ntpdate ntp unzip resolvconf git -y
```

Enable DNS service:

```
systemctl start resolvconf.service
systemctl enable resolvconf.service
systemctl status resolvconf.service
```

Configure DNS service:

```
nano /etc/resolvconf/resolv.conf.d/head
```

Add:

```
nameserver <ip dnsserver domeincontroller>
```

Reload DNS service:

```
systemctl restart resolvconf.service
```

Check if domain controller connection:

```
ping dom001.lan.local
```

Join controller:

```
realm join --user=administrator lan.local --verbose
```

Expected output:

```
* Successfully enrolled machine in realm
```

Edit sssd deamon:

```
nano /etc/sssd/sssd.conf
```

Edit configuration:

```
[sssd]
domains = LAN.LOCAL
config_file_version = 2
services = nss, pam, sudo
default_domain_suffix = lan.local
full_name_format = %1$s

[domain/lan.local]
ad_domain = lan.local
krb5_realm = LAN.LOCAL
realmd_tags = manages-system joined-with-adcli
cache_credentials = True
id_provider = ad
krb5_store_password_if_offline = True
default_shell = /bin/bash
ldap_id_mapping = True
use_fully_qualified_names = True
fallback_homedir = /home/%u@%d
#Disable nested serches (Speeds up searches)
#ignore_group_members = True
#Restict AD search:
#ldap_search_base = DC=lan,DC=local
#ldap_user_search_base OU=Power Users,OU=Accounts,DC=lan,DC=local
#ldap_group_search_base OU=Groups,DC=lan,DC=local
access_provider = simple
simple_allow_groups = <ad group 1>, <ad group 2>
manage-system = yes
automatic-id-mapping = yes
```

Reload sssd deamon:

```
service sssd restart
```

Configure PAM to auto create home folder:

```
nano /etc/pam.d/common-session
```

Add:

```
session    required    pam_mkhomedir.so skel=/etc/skel/ umask=0022
```

Grant root rights (only ubuntu):

```
nano /etc/sudoers
```

Add:

```
%<add ad group here> ALL=(ALL:ALL) ALL
```

To add a additional group use the following command:

```
realm permit -g <groepnaam>@lan.local
```

Secure apache2 login:

```
nano /etc/apache2/sites-enabled/rsx-apache.conf
```

Change the following configuration:

```
#Change in all 3 location blocks:
                Require valid-user
                #Require user user1 user2 user3
#To:
                #Require valid-user
                Require user test01 <<-- username
```

Reload apache2 services:

```
service apache2 restart
```

1.7.2 7.2 Integrate Active Directory LDAP authentication for Apache 2

Activate LDAP module apache:

```
a2enmod ldap authnz_ldap
```

Configure /etc/apache2/apache2.conf as following:

```
<Directory /var/www/html>
AuthType Basic
AuthName "Remote Syslog Login"
Options Indexes FollowSymLinks
AllowOverride None
AuthBasicProvider ldap
AuthLDAPGroupAttributeIsDN On
AuthLDAPURL "ldap://<myadhost>:389/dc=DC01,dc=local?sAMAccountName?sub?(objectClass=*)"
AuthLDAPBindDN "CN=,OU=Accounts,DC=DC01,DC=local"
AuthLDAPBindPassword
AuthLDAPGroupAttribute member
require ldap-group cn=,ou=Groups,dc=DC01,dc=local
</Directory>
```

Reload apache2 services:

```
service apache2 restart
```

1.7.3 7.3 Basic authentication for Apache 2

Install apache2-utils:

```
apt-get install apache2-utils
```

Create .htpasswd file:

```
htpasswd -c /etc/apache2/.htpasswd <myuser>
```

Configure /etc/apache2/apache2.conf as following:

```
<Directory /var/www/html>
AuthType Basic
AuthName "Remote Syslog Login"
AuthBasicProvider file
AuthUserFile "/etc/apache2/.htpasswd"
Require user
Options Indexes FollowSymLinks
AllowOverride None
Require valid-user
Order allow,deny
Allow from all
</Directory>
```

Reload apache2 services:

```
service apache2 restart
```

1.7.4 7.4 Generate an email from an event (Only RSC)

Required core = RSC core

Install netsend:

```
sudo apt install sendmail
```

Edit:

```
/etc/mail/sendmail.cf
```

Search for => #"Smart" relay host (may be null)

Change after DS => DSsmtp.lan.corp

Restart the service:

```
sudo /etc/init.d/sendmail restart
```

Use the following script and save it to /opt/mailrs:

Create array:

```
#!/bin/bash
#Array of words:
declare -a data=(Trace module)
```

Check if error messages exist:

```
for word in "${data[@]}"; do
    mesg=$(cat /var/log/remote_syslog/remote_syslog.log | grep "^$(date +'%b %d')" |␣
↪grep $word)
    if [ -z "$mesg" ]
    then
        echo "No variable!"
    else
```

(continues on next page)

(continued from previous page)

```
        echo "Variable filled, setting variable to continue ..."
        mesgall=1
    fi
done
```

Generate email:

```
if [ -z "$mesgall" ]
then
    echo "Nothing to do, abort"
    exit
else
    echo "Subject: Syslog critical errors" > /opt/rs.txt
    echo "" >> /opt/rs.txt
    echo "Hello <user>," >> /opt/rs.txt
    echo "" >> /opt/rs.txt
    echo "The following message is generated by Remote Syslog." >> /opt/rs.txt
    echo "" >> /opt/rs.txt
    for word in "${data[@]}"; do
        cat /var/log/remote_syslog/remote_syslog.log | grep "^$(date +'%b %d')" | grep
↪$word >> /opt/rs.txt
    done
    echo "" >> /opt/rs.txt
    echo "The messages above are generated by the <hostname>!" >> /opt/rs.txt
    echo "" >> /opt/rs.txt
    echo "Thank you for using Remote Syslog ... ;-)" >> /opt/rs.txt
    cat /opt/rs.txt
    /usr/sbin/sendmail -v -F "T.Slenter" -f "info@mydomain.com" ticketsystem@domain.com
↪< /opt/rs.txt
fi
```

Make file executable:

```
chmod +x /opt/mailrs
```

Install with cron: Command:

```
crontab -e
```

Edit:

```
0 * * * * /opt/mailrs
```

1.7.5 7.5 Generate an email from an event (Only RSE)

Required core = RSE core

Install netsend:

```
sudo apt install sendmail
```

Edit:

```
/etc/mail/sendmail.cf
```

Search for => #"Smart" relay host (may be null)

Change after DS => DSsmtp.lan.corp

Restart the service:

```
sudo /etc/init.d/sendmail restart
```

Use the following script and save it to /opt/mailrs:

Create array:

```
#!/bin/bash
#Array of words:
declare -a data=(module)
```

Check if error messages exist:

```
for word in "${data[@]}"; do
    mesg=$(curl -s -XGET --header 'Content-Type: application/json' http://localhost:9200/
↪rse*/_search -d '{ "query" : { "bool" : { "must": [ { "match": { "MESSAGE": "module" }␣
↪}, { "range": { "@timestamp": { "gte": "now-1h/h" } } } ] } } , "size": 100 }' | /usr/
↪bin/jq | grep $word)
    if [ -z "$mesg" ]
    then
        echo "No variable!"
    else
        echo "Variable filled, setting variable to continue ..."
        mesgall=1
    fi
done
```

Generate email:

```
if [ -z "$mesgall" ]
then
    echo "Nothing to do, abort"
    exit
else
    echo "Subject: Syslog critical errors" > /opt/rs.txt
    echo "" >> /opt/rs.txt
    echo "Hello <user>," >> /opt/rs.txt
    echo "" >> /opt/rs.txt
    echo "The following message is generated by Remote Syslog." >> /opt/rs.txt
```

(continues on next page)

(continued from previous page)

```
    echo "" >> /opt/rs.txt
    for word in "${data[@]}"; do
        curl -s -XGET --header 'Content-Type: application/json' http://localhost:9200/
↪rse*/_search -d '{ "query" : { "bool" : { "must": [ { "match": { "MESSAGE": "module" }␣
↪}, { "range": { "@timestamp": { "gte": "now-1h/h" } } } ] } } , "size": 100 }' | /usr/
↪bin/jq | grep $word >> /opt/rs.txt
    done
    echo "" >> /opt/rs.txt
    echo "The messages above are generated by the <hostname>!" >> /opt/rs.txt
    echo "" >> /opt/rs.txt
    echo "Thank you for using Remote Syslog ... ;-)" >> /opt/rs.txt
    cat /opt/rs.txt
    /usr/sbin/sendmail -v -F "T.Slenter" -f "info@mydomain.com" ticketsystem@domain.com
↪< /opt/rs.txt
fi
```

Make file executable:

```
chmod +x /opt/mailrs
```

Install with cron, run command:

```
crontab -e
```

Edit:

```
0 * * * * /opt/mailrs
```

This will run the script every hour.

1.7.6 7.6 Generate an email from an event using python (Only RSE)

Required core = RSE core

Clone git:

```
git clone https://github.com/tslenter/RSMAILEVENT
```

Edit:

```
./RSMAILEVENT/message.py
```

Change the variable to match the environment.

Make file executable and copy to the opt directory:

```
chmod +x ./RSMAILEVENT/message.py
cp ./RSMAILEVENT/message.py /opt/message.py
```

Install with cron, run command:

```
crontab -e
```

Edit:

```
0 * * * * /opt/message.py
```

This will run the script every hour.

1.8 8. Additional commands

1.8.1 8.1 RSE Core commands

8.1.1 Check the cluster health

Command:

```
curl -XGET -H "Content-Type: application/json" http://localhost:9200/_cluster/health?
↪pretty=true
```

Expected output:

```
{
   "cluster_name" : "rsecore",
   "status" : "green",
   "timed_out" : false,
   "number_of_nodes" : 3,
   "number_of_data_nodes" : 3,
   "active_primary_shards" : 10,
   "active_shards" : 20,
   "relocating_shards" : 0,
   "initializing_shards" : 0,
   "unassigned_shards" : 0,
   "delayed_unassigned_shards" : 0,
   "number_of_pending_tasks" : 0,
   "number_of_in_flight_fetch" : 0,
   "task_max_waiting_in_queue_millis" : 0,
   "active_shards_percent_as_number" : 100.0
}
```

8.1.2 Speed up lifecycly policy check

Command to set it to 1 minute:

```
curl -XPUT --header 'Content-Type: application/json' http://localhost:9200/_cluster/
↪settings --data '
{
   "transient": {
     "indices.lifecycle.poll_interval": "1m"
   }
}'
```

8.1.3 Set lifecycly policy speed to default

Command to reset the policy:

```
curl -XPUT --header 'Content-Type: application/json' http://localhost:9200/_cluster/
↪settings --data '
{
   "transient": {
     "indices.lifecycle.poll_interval": null
   }
}'
```

8.1.4 List indexes

Command to list the indexes:

```
curl -XGET 'localhost:9200/_cat/indices'
```

8.1.5 Check cluster diskspace

Command to list the cluster diskspace:

```
curl -XGET 'localhost:9200/_cat/allocation?v&pretty'
```

8.1.6 Filter on host and time

Adjust size for more results.

Command to filter on host and time:

```
curl -XGET --header 'Content-Type: application/json' http://localhost:9200/rse*/_search -
↪d '{ "query" : { "bool" : { "must": [ { "match": { "HOST_FROM": "172.16.30.1" } }, {
↪"range": { "R_ISODATE": { "gte": "2022-01-13T22:45:39.493+00:00" } } } ] } } , "size":␣
↪3 }' | jq
```

8.1.7 View top 10 results

Command to view top 10 messages:

```
curl -XGET --header 'Content-Type: application/json' http://localhost:9200/rse*/_search?
↪pretty
```

8.1.8 View the mapping of the fields

Command to view mapping of the fields:

```
curl -X GET http://127.0.0.1:9200/rse*/_mapping?pretty
```

8.1.9 Search between times

Adjust size for more results.

Command to view output between a start and top time:

```
curl -XGET --header 'Content-Type: application/json' http://localhost:9200/rse*/_search -
↪d '{ "query" : { "bool" : { "must": [ { "match": { "HOST_FROM": "172.16.30.1" } }, {
↪"range": { "R_ISODATE": { "gte": "2022-01-13T22:45:39.493+00:00", "lte": "2022-01-
↪17T22:45:39.493+00:00" } } } ] } } , "size": 3 }' | jq
```

8.1.10 Search uniq MAC adresses from DHCP index

Command to view output of uniq MAC adresses from a DHCP index:

Requires logstash to index

```
curl -XGET --header 'Content-Type: application/json' http://localhost:9200/logstash-rsx-
↪dhcp*/_search?size=10000 -d '{ "query" : { "bool" : { "should": [ { "match": { "Host_
↪Name": "*NUC00*" } }, { "range": { "@timestamp": { "gte": "now-1d/d" } } } ] } } }' |␣
↪jq | grep MAC_Address | sort | uniq -d
```

8.1.11 View 2 exact terms

Command to view multiple exact terms:

```
curl -X POST http://127.0.0.1:9200/rse*/_search -H 'Content-Type:application/json' -d '{
"query": {
  "terms" : {
    "HOST_FROM" : [ "172.16.30.1", "172.16.30.24" ]
    }
  }
}' | jq
```

8.1.12 View 1 exact term

Command to view 1 exact term:

```
curl -X POST http://127.0.0.1:9200/rse*/_search -H 'Content-Type:application/json' -d  '{
"query": {
  "term" : {
    "HOST_FROM" : "172.16.30.1"
    }
  }
}' | jq
```

8.1.13 Flush indexes

Command to start the flush process of an index makes sure that any data that is currently only persisted in the transaction log is also permanently persisted in Lucene.

```
curl -XPOST --header 'Content-Type: application/json' http://localhost:9200/_flush?wait_
↪if_ongoing | jq
```

or:

```
curl -XPOST --header 'Content-Type: application/json' http://localhost:9200/_flush?wait_
↪if_ongoing | jq
```

Flush a set or a single index:

Note: use wildcard do group the indexes.

```
curl -XPOST --header 'Content-Type: application/json' http://localhost:9200/rse*/_flush␣
↪| jq
```

8.1.14 Delete index

Command to delete a single index:

Index = logstash-rsx-2020.03.28

```
curl -XDELETE http://localhost:9200/logstash-rsx-2020.03.28 | jq
```

8.1.15 View license

Command to view the license:

```
curl -XGET 'http://localhost:9200/_license?pretty'
```

8.1.16 Lite search a value on multiple fields

Command to filter a single value on all fields:

```
curl -XGET 'localhost:9200/_all/_search?q=172.16.30.1&pretty'
```

8.1.17 Lite search a single value for 1 field

Command to filter a single value within 1 field:

```
curl -XGET 'localhost:9200/_all/_search?q=HOST_FROM:172.16.30.1&pretty'
```

8.1.18 Example searches

Create search query for message field:

```
curl -XGET --header 'Content-Type: application/json' http://localhost:9200/rse*/_search -
↪d '{ "query" : { "match" : { "MESSAGE": "172.16.30.1" } } }' | jq
```

or

```
curl -XGET --header 'Content-Type: application/json' http://localhost:9200/rse*/_search -
↪d '{ "query" : { "bool" : { "must": { "match": { "MESSAGE": "172.16.30.1" } } } } }' |␣
↪jq
```

Exclude result based on a single word:

```
curl -XGET --header 'Content-Type: application/json' http://localhost:9200/rse*/_search -
↪d '{ "query" : { "bool" : { "must_not": { "match": { "MESSAGE": "172.16.30.1" } } } } }
↪' | jq
```

8.1.19 Advanced searches

Command to exclude a value and filter down a host within a specific time range:

```
curl -XGET --header 'Content-Type: application/json' http://localhost:9200/rse*/_search -
↪d '{ "query" : { "bool" : { "must_not" : [ { "match" : { "PROGRAM" : "dhcpd" } } ],
↪"filter" : [ { "term": { "HOST_FROM" : "172.16.30.1" } }, { "range": { "R_ISODATE": {
↪"gte": "2022-08-06T10:13:00.000+00:00", "lte": "2022-08-06T10:20:00.000+00:00" } } } ]␣
↪} } , "size": 300 }' | jq
```

Command to filter down a value within a specific time range using OR:

```
curl -XGET --header 'Content-Type: application/json' http://localhost:9200/rse*/_search -
↪d '{ "query" : { "bool" : { "should": [ { "match": { "MESSAGE": "172.16.30.1" } }, {
↪"range": { "R_ISODATE": { "gt": "2022-08-06T10:13:00.000+00:00", "lt": "2022-08-
↪06T10:20:00.000+00:00||+1M" } } } ] } } }' | jq
```

Command to filter down a value within a specific time range using AND (This query uses authentication):

```
curl -XGET --header 'Content-Type: application/json' http://
↪elastic:elastic@localhost:9200/rse*/_search -d '{ "query" : { "bool" : { "must": [ {
↪"match": { "MESSAGE": "marcel" } }, { "range": { "ISODATE": { "gt": "2022-08-
↪12T06:50:14+00:00", "lt": "2022-08-12T06:52:14+00:00" } } } ] } } , "size": 300 }' |␣
↪jq -r -c '.hits.hits[]._source.MESSAGE'
```

Command to exclude a value and filter down multiple hosts within a specific time range:

```
curl -XGET --header 'Content-Type: application/json' http://localhost:9200/rse*/_search -
↪d '{ "query" : { "bool" : { "must_not" : [ { "match" : { "PROGRAM" : "dhcpd" } } ],
↪"filter" : [ { "terms": { "HOST_FROM" : [ "172.16.30.1", "172.16.30.24" ] } }, { "range
↪": { "R_ISODATE": { "gte": "2022-08-06T10:13:00.000+00:00", "lte": "2022-08-
↪06T10:20:00.000+00:00" } } } ] } } , "size": 300 }' | jq
```

Search for value on multiple fields:

Note: Both the fields must match the value.

```
curl -XGET --header 'Content-Type: application/json' http://localhost:9200/rse*/_search -
↪d '{ "query" : { "multi_match" : { "query": "172.16.30.1", "fields": [ "MESSAGE",
↪"HOST_FROM" ] } } }' | jq
```

Search results after data and time with a value using OR:

```
curl -XGET --header 'Content-Type: application/json' http://localhost:9200/rse*/_search -
↪d '{ "query" : { "bool" : { "should": [ { "match": { "MESSAGE": "172.16.30.1" } }, {
↪"range": { "R_ISODATE": { "gte": "2022-08-06T10:13:00.000+00:00" } } } ] } } }' | jq
```

Search results after data and time with a value using AND:

```
curl -XGET --header 'Content-Type: application/json' http://
↪elastic:elastic@localhost:9200/rse*/_search -d '{ "query" : { "bool" : { "must": [ {
↪"match": { "MESSAGE": "marcel" } }, { "match": { "MESSAGE": "VPN" } }, { "range": {
↪"ISODATE": { "gt": "2022-08-12T06:50:14+00:00", "lt": "2022-08-12T06:52:14+00:00" } } }
↪ ] } } , "size": 300 }' | jq -r -c '.hits.hits[]._source.MESSAGE'
```

Search results of the last hour with a value:

```
curl -XGET --header 'Content-Type: application/json' http://localhost:9200/rse*/_search -
↪d '{ "query" : { "bool" : { "should": [ { "match": { "MESSAGE": "172.16.30.1" } }, {
↪"range": { "R_ISODATE": { "gte": "now-1h" } } } ] } } }' | jq
```

8.1.20 Validate query's

Check if query's are valid:

```
curl -XGET --header 'Content-Type: application/json' http://localhost:9200/rse*/_
↪validate/query -d '{ "query" : { "match" : { "MESSAGE": "172.16.30.1" } } }' | jq
```

Check if query is valid with explaination:

```
curl -XGET --header 'Content-Type: application/json' http://localhost:9200/rse*/_
↪validate/query?explain -d '{ "query" : { "match" : { "MESSAGE": "172.16.30.1" } } } }'␣
↪| jq
```

8.1.21 Sort results

Filter value when using sort:

```
curl -XGET --header 'Content-Type: application/json' http://localhost:9200/rse*/_search -
↪d '{ "query" : { "match" : { "MESSAGE": "172.16.30.1" } }, "sort": { "_score": { "order
↪": "desc" } } }' | jq
```

Filter value when using 2 sorts:

```
curl -XGET --header 'Content-Type: application/json' 'http://localhost:9200/rse*/_search?
↪sort=R_ISODATE:desc&sort=_score&q=172.16.30.1' | jq
```

8.1.22 Indexes and aliases

Create a index:

```
curl -XPUT --header 'Content-Type: application/json' http://localhost:9200/rse-dummy | jq
```

Create a alias on a index:

```
curl -XPUT --header 'Content-Type: application/json' http://localhost:9200/rse-dummy/_
↪alias/rse-dummy2 | jq
```

View alias:

```
curl -XGET --header 'Content-Type: application/json' http://localhost:9200/rse-dummy/_
↪alias/* | jq
```

Example alias usage:

```
curl -XDELETE --header "Content-Type: application/json" http://localhost:9200/rsx-
↪netflow*
```

and:

```
curl -XPUT --header 'Content-Type: application/json' http://localhost:9200/rsx-netflow-
↪000001?pretty -d ' { "aliases": { "rsx-netflow":{ "is_write_index": true } } }'
```

8.1.23 Refresh indexes

Refresh all indexes:

```
curl -XPOST --header 'Content-Type: application/json' http://localhost:9200/_refresh | jq
```

Change refresh of index to 30 seconds:

```
curl -XPUT --header 'Content-Type: application/json' http://localhost:9200/rse-dummy/_
↪settings -d '{ "settings": { "refresh_interval": "30s" }}' | jq
```

Disable refresh interval for index:

```
curl -XPUT --header 'Content-Type: application/json' http://localhost:9200/rse-dummy/_
↪settings -d '{ "settings": { "refresh_interval": "-1" }}' | jq
```

Restore default refresh interval for index:

```
curl -XPUT --header 'Content-Type: application/json' http://localhost:9200/rse-dummy/_
↪settings -d '{ "settings": { "refresh_interval": "1s" }}' | jq
```

8.1.24 Example lifecycle policy

```
curl -XPUT --header 'Content-Type: application/json' http://localhost:9200/_ilm/policy/
↪netflow-policy -d ' { "policy": { "phases": { "hot": { "min_age": "0ms", "actions": {
↪"rollover": { "max_primary_shard_size": "50gb", "max_age": "14d" } } }, "delete": {
↪"min_age": "14d", "actions": { "delete": { "delete_searchable_snapshot": true } } } } }
↪ }' | jq
```

and:

```
curl -XPUT --header 'Content-Type: application/json' http://127.0.0.1:9200/_template/
↪netflow-temp -d ' { "template":"rsx-netflow*", "settings": { "number_of_replicas": 1,
↪"number_of_shards": 1, "index.lifecycle.name": "netflow-policy", "index.lifecycle.
↪rollover_alias": "rsx-netflow" } }' | jq
```

8.1.25 Dump latest 10000 results sorted to the CLI

The following is a example commando with authentication. If needed, replace the index and authentication values.

```
curl -XGET --header 'Content-Type: application/json' http://
↪elastic:elastic@localhost:9200/rse*/_search -d '{ "size": 10000, "sort": { "R_ISODATE
↪": "desc"} }' |  jq -r -c '.hits.hits[]._source | "\(.DATE) \(.MESSAGE)"' | tac
```

1.8.2 8.2 RSC Core commands

8.2.1 Search multiple strings of text

```
grep -h "switch1\|switch2\|switch3" /var/log/remote_syslog/* | more
```

8.2.2 Search for the top 15 messages

```
egrep -o "%.+?: "/var/log/remote_syslog/remote_syslog.log | sort | uniq -c | sort -nr |␣
↪head -n 15
```

1.8.3 8.3 Unsupported commands

8.3.1 Disable NTP and change date

```
timedatectl set-time '2022-01-20'
timedatectl set-ntp 0
```

1.9 9. Configuration Files

1.9.1 9.1 Config files locations

9.1.1 Default RSC Core configuration/files

```
Syslog-ng global config:       /etc/syslog-ng/syslog-ng.conf
Syslog-ng additional configs:  /etc/syslog-ng/conf.d/99*
Logrotate:                     /etc/logrotate.d/remotelog
Syslog-ng logrotate:           /etc/logrotate.d/syslog-ng
Colortail global:              /opt/remotesyslog/colortail
```

9.1.2 Default RSE Core configuration/files:

```
Syslog-ng global config:       /etc/syslog-ng/syslog-ng.conf
Syslog-ng additional configs:  /etc/syslog-ng/conf.d/99*
Elasticsearch global config:   /etc/elasticsearch/elasticsearch.yml
```

9.1.3 Default RSX web configuration/files

```
Kibana global config:          /etc/kibana/kibana.yml
```

9.1.4 Default Plugin configuration/files:

```
Filebeat global:               /etc/filebeat/filebeat.yml
Filebeat Cisco:                /etc/filebeat/modules.d/cisco.yml
Filebeat netflow:              /etc/filebeat/modules.d/netflow.yml
Logstash global config:        /etc/logstash/logstash.yml
Logstash additional configs:   /etc/logstash/conf.d/99*
```

1.9.2 9.2 Config checks

9.2.1 Logstash test new config

```
/usr/share/logstash/bin/logstash --config.test_and_exit -f /etc/logstash/conf.d/97-
↪rsmdefault.conf --path.settings /etc/logstash/
```

1.10 10. Security

1.10.1 10.1 Certificate location and replacement

All external connections are encrypted with TLS/SSL, this includes the API on port 8080, SSH and HTTP for user login.

To update the certificates for Apache 2, copy the new certificates to the following directory:

```
/etc/cert/
```

After you copied the new certificates, update the apache2 configuration. File location:

```
/etc/apache2/sites-enabled/
```

Check for the lines:

```
SSLCertificateKeyFile /etc/cert/rs.key
SSLCertificateFile /etc/cert/rs.crt
```

Replace the path with the uploaded certificate.

Reload configuration to apply:

```
service apache2 restart
```

1.10.2 10.2 RS4LOGJ-CVE-2021-44228

Apache Log4j vulnerability - CVE-2021-44228 instructions for Remote Syslog:

Remote Syslog uses the Elasticsearch module to save logging.

Effected products: RSL, RSE and RSX.

RSC is not effected.

10.2.1 Upgrade to recommended version

Check version:

```
curl -XGET 'http://localhost:9200'
```

Output:

```
"number" : "7.16.2"
```

or

```
"number" : "6.8.22"
```

Official document Elasticsearch:

```
https://www.elastic.co/blog/new-elasticsearch-and-logstash-releases-upgrade-apache-log4j2
```

All versions below: 7.16.2 or 6.8.22 are vulnerable. If the version is higher you are good to go and no action is needed.

Upgrade instuction to Elasticsearch 7.16.2 or 6.8.22:

1) In case of a vitual machine create a snapshot
2) Run the upgrade:

```
sudo apt update && sudo apt upgrade
```

!!Please check if the recommended version or higher is going to be installed!!

10.2.2 Mitigation without upgrade

Edit:

```
nano /etc/elasticsearch/jvm.options
```

Add:

```
-Dlog4j2.formatMsgNoLookups=true
```

Restart elasticsearch service:

```
service elasticsearch restart
```

1.11 11. WAF Web Application Firewall

1.11.1 11.1 WAF Requirements

```
Logstash installed
RSE Core
RSX Interface
```

1.11.2 11.2 File download locations

```
Download vcredist_x64.exe: https://www.microsoft.com/en-us/download/details.aspx?id=40784
Download ModSecurityIIS_2.9.3-64b.msi: https://github.com/SpiderLabs/ModSecurity/releases
Download filebeat: https://www.elastic.co/downloads/past-releases/filebeat-7-17-3
Download winlogbeat: https://www.elastic.co/downloads/beats/winlogbeat
Download configuration: https://github.com/tslenter/RSWAFCONF
```

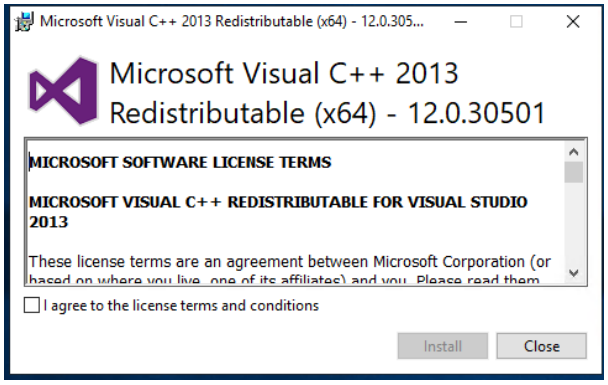
!Notice!: filebeat version 8 > is not working with the this guide.

1.11.3 11.3 IIS Module installation

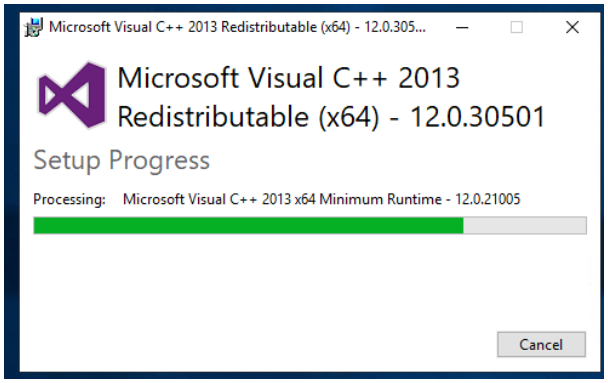
11.3.1 Visual C++ Redistributable Packages installation

Step 1

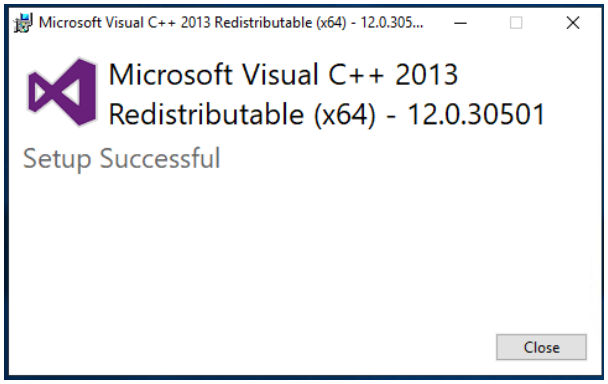
Start the installation of the Visual C++ Redistributable packages.



Accept and click Install.



Wait for the installation to finish.



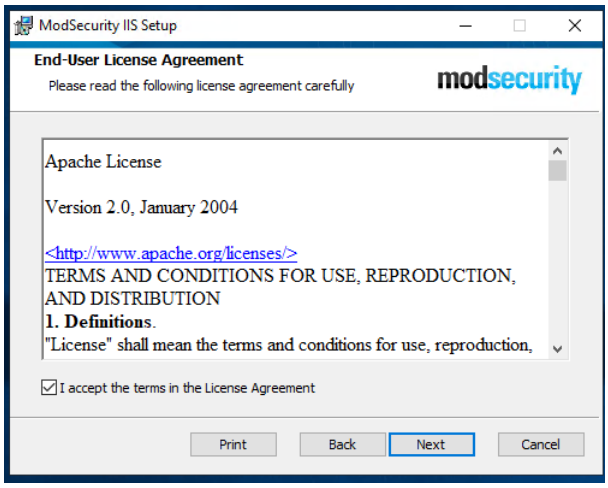
Click close. The installation is done.

11.3.2 ModSecurity installation

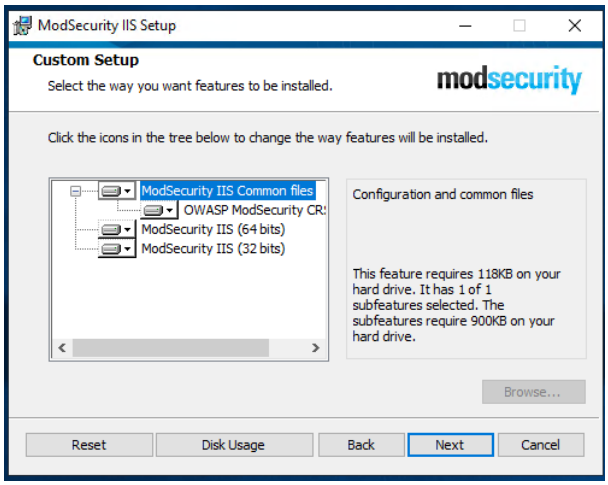
Step 2

Start the installation of the ModSecurity package.

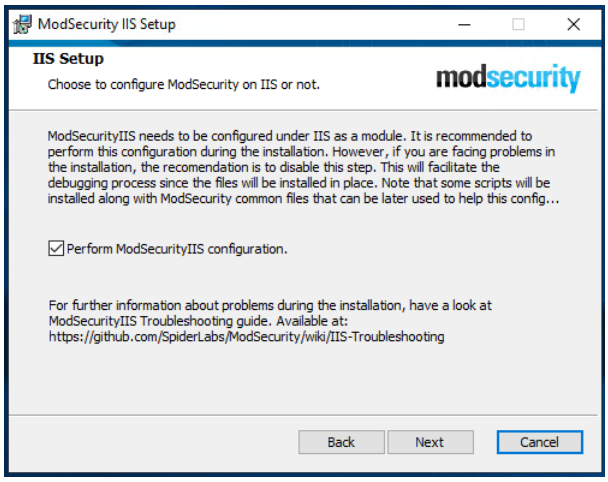
Click next.



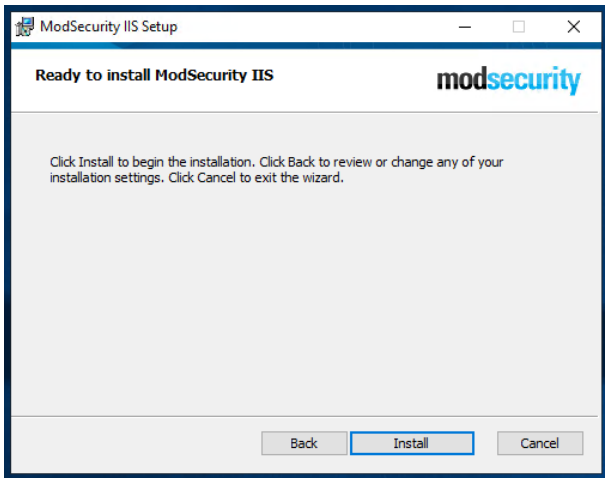
Accept and click next.



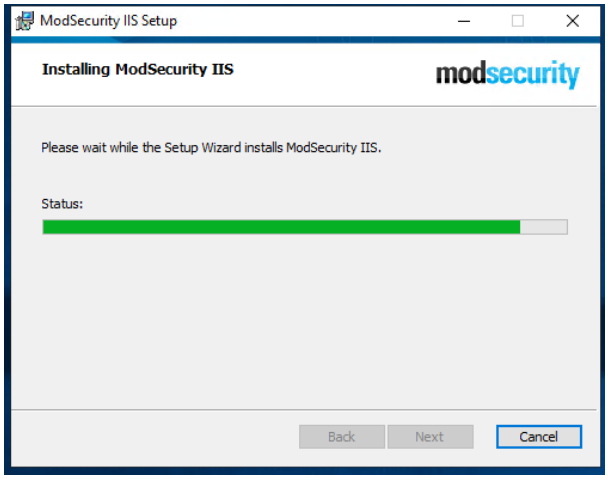
Click next.



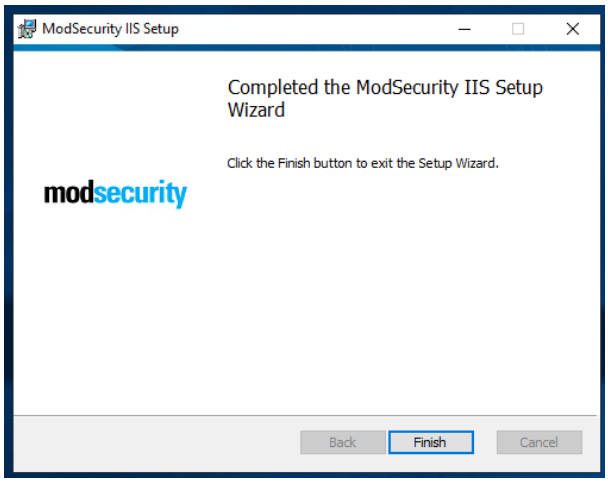
Click next.



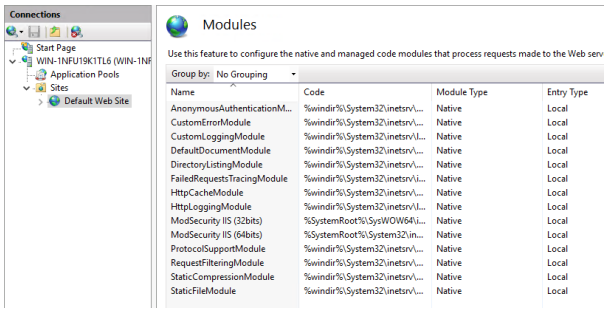
Click Install.



Wait for the installation to finish.



Click Finish.



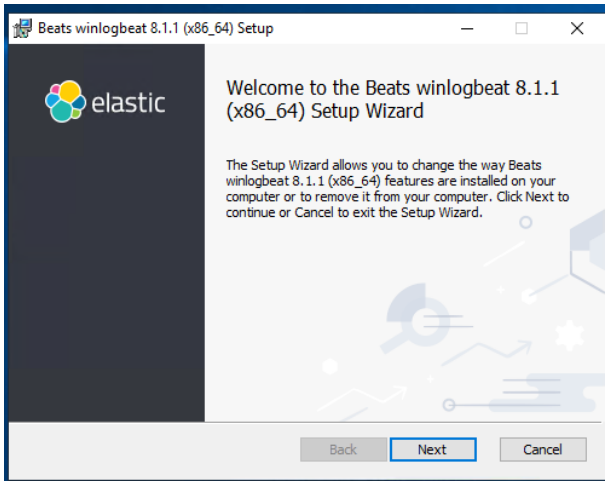
Check within the IIS console if the modules are loaded.

Depending of the installation go to section 11.3.3 (WinLogBeat) or 11.3.4 (Filebeat).

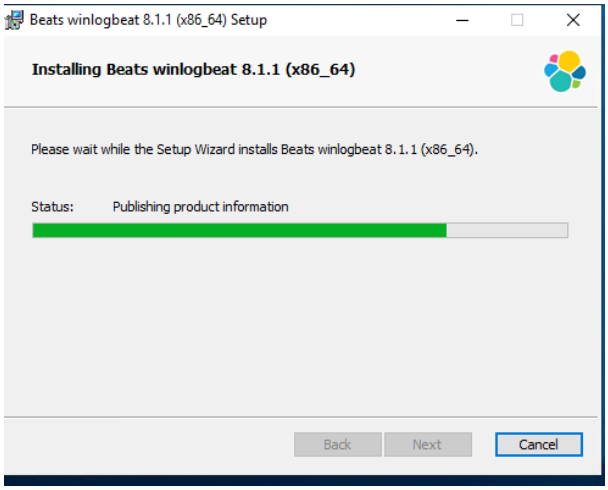
11.3.3 WinLogBeat installation

Step 3

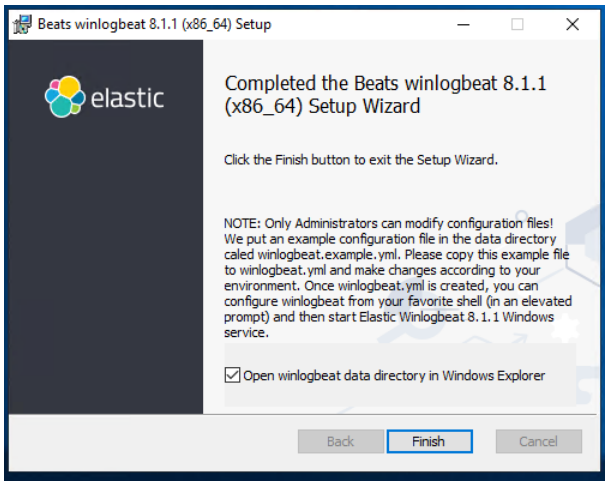
Start the installation of the ModSecurity package.



Accept and click Install.



Wait for the installation to finish.

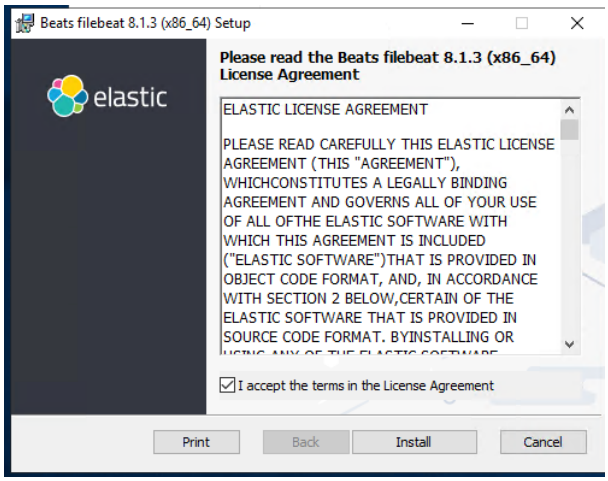


Click Finish.

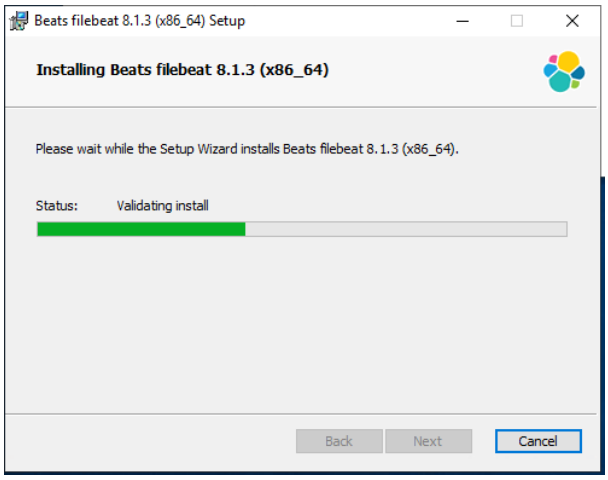
11.3.4 Filebeat installation

Step 3

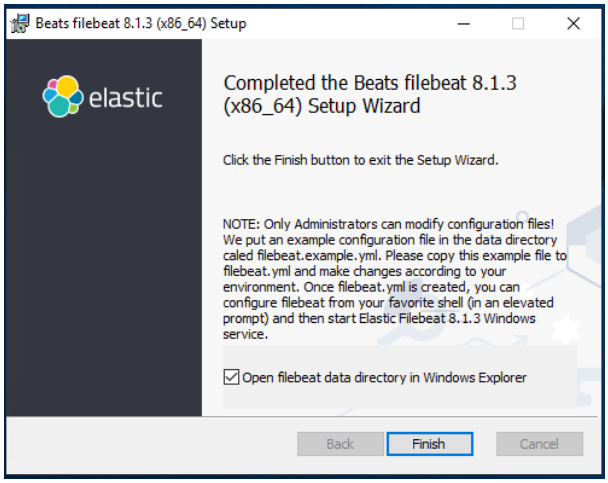
Start the installation of the ModSecurity package.



Accept and click Install.



Wait for the installation to finish.



Click Finish.

11.3.5 ModSecurity Configuration

Step 4

Check the RSWAFCONF git for the MODSECURITY folder and copy all files to:

```
C:\Program Files\ModSecurity IIS
```

Edit modsecurity.conf (optional):

```
# based on modsecurity.conf-recommended
# -- Rule engine initialization ----------------------------------------------

# Enable ModSecurity, attaching it to every transaction. Use detection
# only to start with, because that minimises the chances of post-installation
# disruption.
#
#SecRuleEngine DetectionOnly
SecRuleEngine On

# -- Request body handling ---------------------------------------------------

# Allow ModSecurity to access request bodies. If you don't, ModSecurity
# won't be able to see any POST parameters, which opens a large security
# hole for attackers to exploit.
#
SecRequestBodyAccess On

# SecStreamInBodyInspection is required by IIS for proper body inspection
# See issue #1299 for more information
SecStreamInBodyInspection On

# Enable XML request body parser.
# Initiate XML Processor in case of xml content-type
#
SecRule REQUEST_HEADERS:Content-Type "(?:application(?:/soap\+|/)|text/)xml" \
         "id:'200000',phase:1,t:none,t:lowercase,pass,nolog,ctl:requestBodyProcessor=XML"

# Enable JSON request body parser.
# Initiate JSON Processor in case of JSON content-type; change accordingly
# if your application does not use 'application/json'
#
SecRule REQUEST_HEADERS:Content-Type "application/json" \
         "id:'200001',phase:1,t:none,t:lowercase,pass,nolog,ctl:requestBodyProcessor=JSON
↪"

# Maximum request body size we will accept for buffering. If you support
# file uploads then the value given on the first line has to be as large
# as the largest file you are willing to accept. The second value refers
# to the size of data, with files excluded. You want to keep that value as
# low as practical.
#
SecRequestBodyLimit 13107200
SecRequestBodyNoFilesLimit 131072
```

(continues on next page)

(continued from previous page)

```
# Store up to 128 KB of request body data in memory. When the multipart
# parser reaches this limit, it will start using your hard disk for
# storage. That is slow, but unavoidable.
#
SecRequestBodyInMemoryLimit 131072

# What do do if the request body size is above our configured limit.
# Keep in mind that this setting will automatically be set to ProcessPartial
# when SecRuleEngine is set to DetectionOnly mode in order to minimize
# disruptions when initially deploying ModSecurity.
#
SecRequestBodyLimitAction Reject

# Verify that we've correctly processed the request body.
# As a rule of thumb, when failing to process a request body
# you should reject the request (when deployed in blocking mode)
# or log a high-severity alert (when deployed in detection-only mode).
#
SecRule REQBODY_ERROR "!@eq 0" \
"id:'200002', phase:2,t:none,log,deny,status:400,msg:'Failed to parse request body.',
↪logdata:'%{reqbody_error_msg}',severity:2"

# By default be strict with what we accept in the multipart/form-data
# request body. If the rule below proves to be too strict for your
# environment consider changing it to detection-only. You are encouraged
# _not_ to remove it altogether.
#
SecRule MULTIPART_STRICT_ERROR "!@eq 0" \
"id:'200003',phase:2,t:none,log,deny,status:400, \
msg:'Multipart request body failed strict validation: \
PE %{REQBODY_PROCESSOR_ERROR}, \
BQ %{MULTIPART_BOUNDARY_QUOTED}, \
BW %{MULTIPART_BOUNDARY_WHITESPACE}, \
DB %{MULTIPART_DATA_BEFORE}, \
DA %{MULTIPART_DATA_AFTER}, \
HF %{MULTIPART_HEADER_FOLDING}, \
LF %{MULTIPART_LF_LINE}, \
SM %{MULTIPART_MISSING_SEMICOLON}, \
IQ %{MULTIPART_INVALID_QUOTING}, \
IP %{MULTIPART_INVALID_PART}, \
IH %{MULTIPART_INVALID_HEADER_FOLDING}, \
FL %{MULTIPART_FILE_LIMIT_EXCEEDED}'"

# Did we see anything that might be a boundary?
#
SecRule MULTIPART_UNMATCHED_BOUNDARY "!@eq 0" \
"id:'200004',phase:2,t:none,log,deny,msg:'Multipart parser detected a possible unmatched␣
↪boundary.'"

# PCRE Tuning
# We want to avoid a potential RegEx DoS condition
```

(continues on next page)

(continued from previous page)

```
#
SecPcreMatchLimit 1000
SecPcreMatchLimitRecursion 1000

# Some internal errors will set flags in TX and we will need to look for these.
# All of these are prefixed with "MSC_".  The following flags currently exist:
#
# MSC_PCRE_LIMITS_EXCEEDED: PCRE match limits were exceeded.
#
SecRule TX:/^MSC_/ "!@streq 0" \
                "id:'200005',phase:2,t:none,deny,msg:'ModSecurity internal error␣
↪flagged: %{MATCHED_VAR_NAME}'"


# -- Response body handling ---------------------------------------------------

# Allow ModSecurity to access response bodies.
# You should have this directive enabled in order to identify errors
# and data leakage issues.
#
# Do keep in mind that enabling this directive does increases both
# memory consumption and response latency.
#
SecResponseBodyAccess On

# Which response MIME types do you want to inspect? You should adjust the
# configuration below to catch documents but avoid static files
# (e.g., images and archives).
#
SecResponseBodyMimeType text/plain text/html text/xml

# Buffer response bodies of up to 512 KB in length.
SecResponseBodyLimit 524288

# What happens when we encounter a response body larger than the configured
# limit? By default, we process what we have and let the rest through.
# That's somewhat less secure, but does not break any legitimate pages.
#
SecResponseBodyLimitAction ProcessPartial

# -- Filesystem configuration -------------------------------------------------

# The location where ModSecurity stores temporary files (for example, when
# it needs to handle a file upload that is larger than the configured limit).
#
# This default setting is chosen due to all systems have /tmp available however,
# this is less than ideal. It is recommended that you specify a location that's private.
#
SecTmpDir c:\inetpub\temp\

# The location where ModSecurity will keep its persistent data.  This default setting
# is chosen due to all systems have /tmp available however, it
```

(continues on next page)

(continued from previous page)

```
# too should be updated to a place that other users can't access.
#
SecDataDir c:\inetpub\temp\

# -- File uploads handling configuration ---------------------------------

# The location where ModSecurity stores intercepted uploaded files. This
# location must be private to ModSecurity. You don't want other users on
# the server to access the files, do you?
#
#SecUploadDir c:\inetpub\temp\

# By default, only keep the files that were determined to be unusual
# in some way (by an external inspection script). For this to work you
# will also need at least one file inspection rule.
#
#SecUploadKeepFiles RelevantOnly

# Uploaded files are by default created with permissions that do not allow
# any other user to access them. You may need to relax that if you want to
# interface ModSecurity to an external program (e.g., an anti-virus).
#
#SecUploadFileMode 0600

# -- Debug log configuration ---------------------------------------------

# The default debug log configuration is to duplicate the error, warning
# and notice messages from the error log.
#
#SecDebugLog c:\inetpub\temp\debug.log
#SecDebugLogLevel 3

# -- Audit log configuration ---------------------------------------------

# Log the transactions that are marked by a rule, as well as those that
# trigger a server error (determined by a 5xx or 4xx, excluding 404,
# level response status codes).
#
SecAuditEngine RelevantOnly
SecAuditLogRelevantStatus "^(?:5|4(?!04))"

# Log everything we know about a transaction.
SecAuditLogParts ABIJDEFHZ
SecAuditLogFormat JSON

# Use a single file for logging. This is much easier to look at, but
# assumes that you will use the audit log only ocassionally.
#
SecAuditLogType Serial
SecAuditLog D:\MOD-Security_LOG\modsec_audit.log

# Specify the path for concurrent audit logging.
```

(continues on next page)

(continued from previous page)

```
SecAuditLogStorageDir C:\MOD-Security_LOG

# -- Miscellaneous ------------------------------------------------------

# Use the most commonly used application/x-www-form-urlencoded parameter
# separator. There's probably only one application somewhere that uses
# something else so don't expect to change this value.
#
SecArgumentSeparator &

# Settle on version 0 (zero) cookies, as that is what most applications
# use. Using an incorrect cookie version may open your installation to
# evasion attacks (against the rules that examine named cookies).
#
SecCookieFormat 0

# Specify your Unicode Code Point.
# This mapping is used by the t:urlDecodeUni transformation function
# to properly map encoded data to your language. Properly setting
# these directives helps to reduce false positives and negatives.
#
SecUnicodeMapFile unicode.mapping 20127

# Improve the quality of ModSecurity by sharing information about your
# current ModSecurity version and dependencies versions.
# The following information will be shared: ModSecurity version,
# Web Server version, APR version, PCRE version, Lua version, Libxml2
# version, Anonymous unique id for host.
SecStatusEngine On
```

Check the crs-setup.conf.example (Optional):

Make sure the following paranoia level is set (Optional:

```
SecAction \
  "id:900000,\
   phase:1,\
   nolog,\
   pass,\
   t:none,\
   setvar:tx.paranoia_level=2"
```

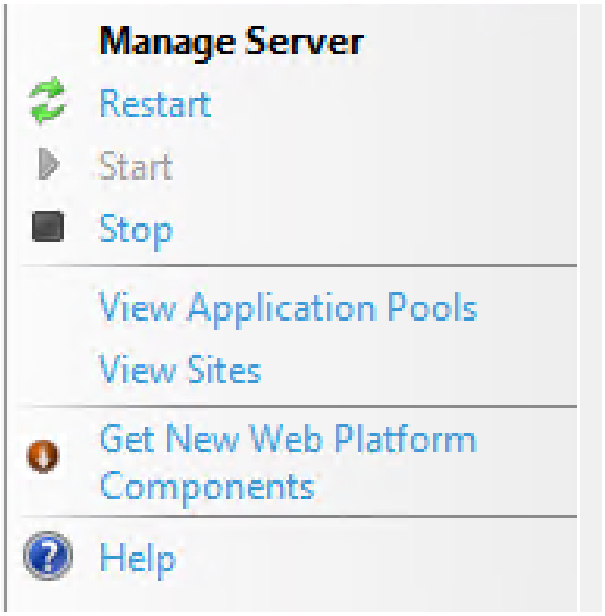
Greate the folowing directory:

```
C:\MOD-Security_LOG
```

Run:

```
cacls C:\inetpub\temp /e /p IIS_IUSRS:f
cacls C:\MOD-Security_LOG /e /p IIS_IUSRS:f
```

Reload the IIS service:



Click restart.

Mod security is now installed. By default we block on the OWASP ruleset. If you only want to monitor change within the modsecurity.conf the following code (Optional):

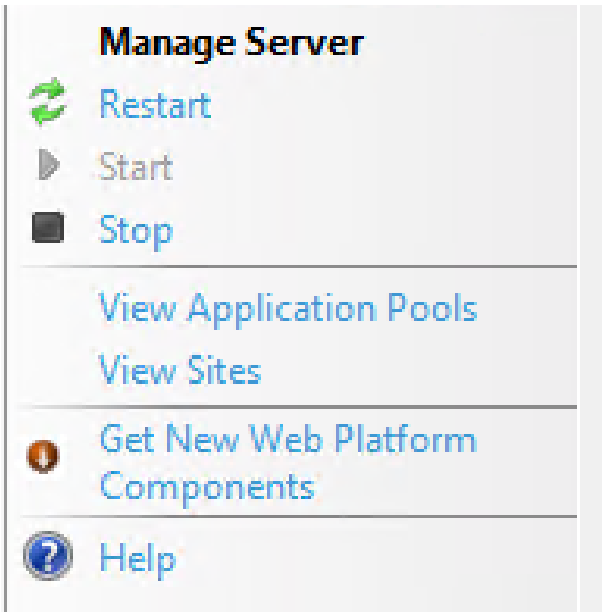
From:

```
#SecRuleEngine DetectionOnly
SecRuleEngine On
```

To:

```
SecRuleEngine DetectionOnly
#SecRuleEngine On
```

If the detection mode is changed do a reload of the service (reload from the IIS console):



To prevent a big “modsec_audit.log” create a batch file and schedule it 1 or 2 times a day. Example (Optional):

```
@echo off
IISReset /STOP
del "c:\MOD-Security_LOG\modsec_audit.log"
IISReset /START
```

Example file location (Optional):

```
c:\CLEAR_MOD_SEC_LOGGING.bat
```

If you run WinLogBeat you can disable the following configuration within the “modsecurity.conf” (Optional):

```
#SecAuditLog D:\MOD-Security_LOG\modsec_audit.log
#SecAuditLogStorageDir C:\MOD-Security_LOG
```

A batch file is not needed if the configuration of the log file is disabled using #.

11.3.6 or 11.3.7 can be followed as step 5.

11.3.6 WinLogBeat Configuration

Step 5

Configuration of the WinLogBeat package.

Go to the following directory:

```
C:\ProgramData\Elastic\Beats\winlogbeat
```

Edit the winlogbeat.yml:

```
winlogbeat.event_logs:
  - name: Application
    ignore_older: 72h
    provider:
     - ModSecurity

setup.template.settings:
  index.number_of_shards: 1

output.logstash:
  # The Logstash hosts
  hosts: ["cloud.remotesyslog.com:22222"]

processors:
  - add_host_metadata: ~
  - add_cloud_metadata: ~
setup.template.fields: ${path.config}/fields.yml
setup.template.json.enabled: false
setup.template.overwrite: true
```

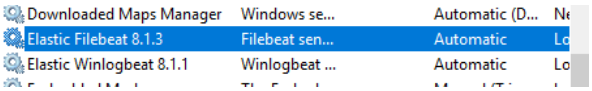
A Example can found here:

```
https://github.com/tslenter/RSWAFCONF/tree/main/WINLOGBEAT
```

Replace the field.yml with the file given in the following URL:

```
https://github.com/tslenter/RSWAFCONF/tree/main/WINLOGBEAT
```

Reload the WinLogBeat service:



On the server side (Logstash with the RSE Core) add the following configuration:

Create and edit a file:

```
nano /etc/logstash/conf.d/99-myprogram.conf
```

Add the following configuration:

```
input {
  beats {
    port => 22222
  }
}

filter {
  mutate {
     rename => { "[winlog][event_data][param1]" => "message" }
  }
  mutate { gsub => [ "message", ".*ModSecurity: [^\[]+\[", "" ] }
  mutate { gsub => [ "message", "][^\[]+$", "" ] }
  kv { field_split_pattern => "] \[" value_split => " " }
#  dissect { mapping => { "message" => "[%{[@metadata][timestamp]}]%{}" } }
#  date { match => [ "[@metadata][timestamp]", "EEE MMM dd HH:mm:ss.SSSSSS yyyy" ] }
}


output {
 if [host][hostname] == "SENDING_SERVER" {
  elasticsearch { hosts => ["localhost:9200"] index => "rse-myprogram"
    }
 }
stdout { codec => rubydebug }
```

Change “SENDING_SERVER” in the hostname of your host which sends logging.

11.3.7 FileBeat Configuration

Step 5

Configuration of the Filebeat package.

Go to the following directory:

```
C:\ProgramData\Elastic\Beats\filebeat
```

Download the FileBeat configuration with the URL below and override the files within the FileBeat configuration folder. Expect for the modules folder.

```
https://github.com/tslenter/RSWAFCONF/tree/main/FILEBEAT
```

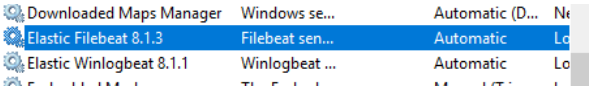
Copy the module directory to:

```
C:\Program Files\Elastic\Beats\<version>\filebeat
```

Edit the filebeat.yml file with the server information:

```
output.logstash:
   hosts: ["cloud.remotesyslog.com:22222"]
#  Enable if CA is enabled
#  ssl.enabled: true
#  ssl.certificate_authorities: ["${path.config}/cacert.crt"]
```

Reload the Filebeat service:



On the server side (Logstash with the RSE Core) add the following configuration:

Create and edit the following file:

```
nano /etc/logstash/conf.d/99-myprogram.conf
```

Add the following configuration:

```
input {
  beats {
    port => 22222
  }
}

#use with filebeat
filter {
   json {
      source => "message"
   }
}

output {
 if [host][hostname] == "SENDING_SERVER" {
  elasticsearch { hosts => ["localhost:9200"] index => "rse-myprogram"
    }
 }
stdout { codec => rubydebug }
```

Change “SENDING_SERVER” in the hostname of your host which sends logging.

1.12 12. FAQ

1.12.1 12.1 My RSX/RSE installation does not recieve any logging

You probably should check the date. If the date is not correct run in the CLI as root:

```
dpkg-reconfigure tzdata
```

This allows you to configure the timezone.

The next thing to check is within the Kibana console: Management => Advanced Settings => Timezone for date formatting => setup the right timezone.

1.12.2 12.2 Disk full by Geo2

Message in logging:

```
Jan 27 10:24:50 plisk002.prd.corp syslog-ng[1793]: geoip2(): getaddrinfo failed; gai_
↪error='Name or service not known', ip='', location='/etc/syslog-ng/conf.d/99X-
↪Checkpoint.conf:32:25'
Jan 27 10:24:50 plisk002.prd.corp syslog-ng[1793]: geoip2(): maxminddb error; error=
↪'Unknown error code', ip='', location='/etc/syslog-ng/conf.d/99X-Checkpoint.conf:32:25'
```

Components needed for fix:

```
File: /etc/syslog-ng/syslog-ng.conf
```

File destinations:

```
- d_syslog
- d_error
```

Log rules:

```
- log { source(s_src); filter(f_syslog3); destination(d_syslog); };
- log { source(s_src); filter(f_error); destination(d_error); };
```

Fix:

```
vi /etc/syslog-ng/syslog-ng.conf
```

Add rules:

```
filter geoip_messages_1 { not match("Name or service not known"); };
filter geoip_messages_2 { not match("Unknown error code"); };
```

Change rules:

```
-log { source(s_src); filter(f_syslog3); destination(d_syslog); };
-log { source(s_src); filter(f_error); destination(d_error); };
+log { source(s_src); filter(f_syslog3); filter(geoip_messages_1); filter(geoip_messages_
↪2); destination(d_syslog); };
+log { source(s_src); filter(f_error); filter(geoip_messages_1); filter(geoip_messages_
↪2); destination(d_error); };
```

1.12.3 12.3 Kibana not loaded after upgrade

Restarting the server will solve this problem. Some report that a restart of the Kibana or Elasticsearch will fix the issue.

```
service elasticsearch restart
service kibana restart
```

1.12.4 12.4 Data too large, data for [<http_request>] (JVM heap size)

Error message:

```
tom@plisx002:~$ curl -X GET 'http://localhost:9200/_cat/health?v'
{"error":{"root_cause":[{"type":"circuit_breaking_exception","reason":"[parent] Data too␣
↪large, data for [<http_request>] would be [1014538592/967.5mb], which is larger than␣
↪the limit of [986061209/940.3mb], real usage: [1014538592/967.5mb], new bytes␣
↪reserved: [0/0b], usages [request=0/0b, fielddata=3057213/2.9mb, in_flight_requests=0/
↪0b, accounting=261018719/248.9mb]","bytes_wanted":1014538592,"bytes_limit":986061209,
↪"durability":"PERMANENT"}],"type":"circuit_breaking_exception","reason":"[parent] Data␣
↪too large, data for [<http_request>] would be [1014538592/967.5mb], which is larger␣
↪than the limit of [986061209/940.3mb], real usage: [1014538592/967.5mb], new bytes␣
↪reserved: [0/0b], usages [request=0/0b, fielddata=3057213/2.9mb, in_flight_requests=0/
↪0b, accounting=261018719/248.9mb]","bytes_wanted":1014538592,"bytes_limit":986061209,
↪"durability":"PERMANENT"},"status":429}
```

Increase memory fix:

```
nano /etc/elasticsearch/jvm.options
```

Edit:

```
--Xms1g
--Xmx1g
+-Xms6g
+-Xmx6g
```

1.12.5 12.5 Syslog-NG 3.27.1 breaks with new upgrade on Ubuntu 18.04 and 20.04

Error message:

```
dpkg: error processing package syslog-ng-mod-sql (--configure):
 dependency problems - leaving unconfigured
dpkg: dependency problems prevent configuration of syslog-ng-mod-redis:
 syslog-ng-mod-redis depends on syslog-ng-core (>= 3.27.1-2); however:
  Package syslog-ng-core is not configured yet.
 syslog-ng-mod-redis depends on syslog-ng-core (<< 3.27.1-2.1~); however:
  Package syslog-ng-core is not configured yet.
```

Fix:

Backup configuration:

```
mkdir ~/syslog-ng_backup/
cp -rf /etc/syslog-ng/* ~/syslog-ng_backup/
```

Verify configuration:

```
ls ~/syslog-ng_backup/
```

Purge syslog-ng and remove everything:

```
sudo apt purge syslog-ng-core
```

If some files remain, delete them all:

```
rm -rf /etc/syslog-ng
```

Reinstall syslog-ng-core:

```
sudo apt install syslog-ng-core
```

Reinstall syslog-ng:

```
sudo apt install syslog-ng
```

Cleanup some packages:

```
sudo apt auto-remove
```

Restore RS configuration files:

```
cp ~/syslog-ng_backup/conf.d/99* /etc/syslog-ng/conf.d/
```

If you edited the /etc/syslog-ng/syslog-ng.conf file, check the difference and restore your custom configuration.

This issue should be fixed in version 3.27.1-2.1 and higher.

1.12.6 12.6 How to resolve a full disk

Error recieved in Kibana:

```
{"type":"log","@timestamp":"2022-08-29T15:55:49+02:00","tags":["info","savedobjects-
↪service"],"pid":8508,"message":"[.kibana_task_manager] REINDEX_SOURCE_TO_TEMP_
↪TRANSFORM -> REINDEX_SOURCE_TO_TEMP_INDEX_BULK. took: 64ms."}
{"type":"log","@timestamp":"2022-08-29T15:55:49+02:00","tags":["info","savedobjects-
↪service"],"pid":8508,"message":"[.kibana] REINDEX_SOURCE_TO_TEMP_TRANSFORM -> REINDEX_
↪SOURCE_TO_TEMP_INDEX_BULK. took: 124ms."}
```

Run:

```
curl -XGET -H "Content-Type: application/json" http://elastic:elastic@localhost:9200/_
↪cluster/allocation/explain?pretty
```

Given warning:

```
"explanation" : "the node is above the low watermark cluster setting [cluster.routing.
↪allocation.disk.watermark.low=85%], using more disk space than the maximum allowed [85.
↪0%], actual free: [13.201362304896152%]"
```

Temporary increase the allowed diskspace:

```
curl -XPUT -H "Content-Type: application/json" http://elastic:elastic@localhost:9200/_
↪cluster/settings -d '
{
  "transient": {
    "cluster.routing.allocation.disk.watermark.low": "90%",
    "cluster.routing.allocation.disk.watermark.high": "92%",
    "cluster.routing.allocation.disk.watermark.flood_stage": "95%"
  }
}' | jq
```

Login and remove some shards to lower the diskspace then restore the allowed diskspace:

```
curl -XPUT -H "Content-Type: application/json" http://elastic:elastic@localhost:9200/_
↪cluster/settings -d '
{
  "transient": {
    "cluster.routing.allocation.disk.watermark.low": "85%",
    "cluster.routing.allocation.disk.watermark.high": "90%",
    "cluster.routing.allocation.disk.watermark.flood_stage": "95%"
  }
}' | jq
```